NCAR/TN-435-STR
NCAR TECHNICAL NOTE

July 1997

Introduction to Adjoint Techniques and the MM5 Adjoint Modeling System

Xiaolei Zou
F. Vandenberghe
Manuel Pondeca
Y.-H. Kuo

MESOSCALE AND MICROSCALE METEORLOGY DIVISION

NATIONAL CENTER FOR ATMOSPHERIC RESEARCH
BOULDER, COLORADO

NCAR TECHNICAL NOTES

The Technical Note series provides an outlet for a variety of NCAR Manuscripts that contribute in specialized ways to the body of scientific knowledge but that are not suitable for journal, monograph, or book publication. Reports in this series are issued by the NCAR scientific divisions. Designation symbols for the series include:

EDD—Engineering, Design, or Development Reports
Equipment descriptions, test results, instrumentation, and operating and maintenance manuals.

IA—Instructional Aids
Instruction manuals, bibliographies, film supplements, and other research or instructional aids.

PPR-Program Progress Reports
Field program reports, interim and working reports, survey reports, and plans for experiments.

PROC—Proceedings
Documentation or symposia, colloquia, conferences, workshops, and lectures. (Distribution may be limited to attendees.)

STR-Scientific and Technical Reports
Data compilations, theoretical and numerical investigations, and experimental results.

The National Center for Atmospheric Research (NCAR) is operated by the nonprofit University Corporation for Atmospheric Research (UCAR) under the sponsorship of the National Science Foundation. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the authors(s) and do not necessarily reflect the views of the National Science Foundation.

INTRODUCTION TO ADJOINT TECHNIQUES AND THE MM5 ADJOINT MODELING SYSTEM

Xiaolei Zou, F. Vandenberghe, M. Pondeca, and Y.-H. Kuo

NCAR Technical Note, 1997, NCAR/TN-435-STR

TABLE OF CONTENTS

Preceding page blank

List of Figures

PREFACE

This technical report, revised in the summer of 1998, introduces the concept of adjoint from its various applications in meteorology and describes the MM5 adjoint modeling system. It is intended to provide scientific and technical documentation of the system for users who will use such a system as one of their research tools. Comments and suggestions for improvements or corrections are welcome and should be sent to the authors. Users who want to know more about the overall MM5 adjoint modeling system should obtain a copy of *A User's Guide to the MM5 Adjoint Modeling System*, by Zou, Huang and Xiao (1998).

ACKNOWLEDGMENTS

A number of scientists, students, and sponsors have made significant contributions towards developing this system. We would like to thank Y.-R. Guo, W. Huang, T.-Y. Xu, and T. Jian who directly contributed to this development effort. Special thank goes to J. Dudhia for his extreme patience in answering all the questions we had for the MM5 forward model related to the adjoint model development.

We would also like to extend our gratitude to the two reviewers (Drs. J. Dudhia and B. Snyder), whose comments improved the manuscript significantly.

Computing for this model development was undertaken on the CRAY YMP and Cray J90 at NCAR, which is supported by the National Science Foundation.

This effort is jointly supported by the National Science Foundation, the Federal Aviation Administration through a corporate agreement with NCAR and UCAR, ATM9209181 under scientific program order EAR9405501, the Department of Energy's Atmospheric Radiation Measurement Program (DOE/ARM) grant number DE-A105-90ER1070, and the AFOSR through project number 95NM373.

Finally, we especially thank Bob Gall for his strong and steadfast support during the entire period of this development.

ABSTRACT

Adjoint models are increasingly applied to many problems in meteorology and oceanography. The adjoint model of MM5 is a tool which effectively computes the gradient (or a Gateau-derivative) of any MM5 forecast aspect with respect to the model's control variables, which consist of model initial conditions, boundary conditions, and model parameters that define the physical and numerical conditions of the integration.

Different applications of adjoint models in meteorology are briefly reviewed and their mathematical formulae are provided which illustrate how the adjoint model and/or tangent linear model are used in each application. Then we describe the mathematical and numerical formulation used in developing the adjoint version of MM5. The possibility of carrying out optimal control of lateral boundary condition in addition to the initial condition, the restart of minimization procedure, the proper handling of disk space for large problems, and the choice of different basic state update frequencies are provided. Finally, problems that might arise in the practical coding of the adjoint of a numerical model are summarized. A number of rules for the practical coding of tangent linear and adjoint models, along with various examples, are presented. The problems raised by the development and maintenance of adjoint codes are also discussed.

CHAPTER 1. INTRODUCTION

An ideal assimilation process must use all the available information i.e., the observations, the physical laws governing the flow, and the known statistical properties of the flow to produce a complete and consistent description of the flow with its associated uncertainty. Estimation theory provides a general conceptual framework and a number of algorithms for solving this problem. However, estimation theory has been developed primarily for engineering problems which are in general linear and of low dimension. Atmospheric data assimilation problems are generally nonlinear and of large dimension. Computational approximations must, therefore, be developed to efficiently implement these algorithms for atmospheric and oceanic problems. Indeed, a large amount of the work in data assimilation has been devoted to the development of some cost effective simplifications to known algorithms of estimation theory. An assimilation process involves various aspects of atmospheric science, statistics, computer science, instrumental physics, and control theory.

Control theory has permitted probably the most significant progress accomplished over last 10 years in data assimilation research. It offers a deterministic approach of the estimation problem posed by data assimilation. In this approach, variational methods are used to formulate the data assimilation problem as an optimization problem, which can then be tackled using classical numerical methods, and is much easier to handle than the stochastic calculus of the estimation theory. The introduction of the so-called adjoint techniques, has dramatically reduced the computational expenses and has opened new horizons for data assimilation and other research areas in meteorology and oceanography. The adjoint techniques are the central topic of this report and will be extensively described. Details of the development of the adjoint models, with specific emphasis on the MM5 adjoint model, will be provided.

The documentation is divided into three parts. Part I provides a general description of the adjoint technique and its various applications, Part II presents the MM5 adjoint model system, and Part III summarizes the practical adjoint coding experiences.

CHAPTER 2: A GENERAL DESCRIPTION OF ADJOINT TECHNIQUES

2.1 A simple statistical estimation problem

We introduce some of the basic probability concepts underlying the data assimilation problem.

Let's consider a simple estimation problem (Talagrand, 1992). Suppose a parameter x has been measured at the same location and same time with two different methods. We use y_1 and σ_1 (y_2 and σ_2) to represent the measured value of x and its measurement uncertainty of the first (second) measurement. For instance, we can assume that x is the temperature of an object, y_1 is the value obtained with a certain measurement device of accuracy $\sigma_1 = 0.5^\circ\ C$, and y_2 is the value measured with a different device of accuracy $\sigma_2 = 1.0^\circ\ C$. From these two measurements, one wants to obtain an estimate of the actual temperature with its associated error.

A naive idea would be to retain the first measurement which comes from a more accurate device and neglect the second which is less accurate. This is not always true since both measurements are realizations of a random process and, in some (unlikely) extreme situations, measurement 2 (y_2) can be closer to the true value than the measurement 1 (y_1) is. It is better on average to use both observations to derive an estimate of x in order to reduce the measurement uncertainty. This is the purpose of the estimation theory.

To present this solution mathematically, some basic probability concepts must be introduced. Measurements y_1 and y_2 can be seen as realizations of two independent random variables[1] Y_1 and Y_2 (see Jazwinsky 1970, chapter 2). If we assume that measurements are unbiased, i.e., when measurements are infinitely repeated they produce an average value which equals to x, we can assign a particular form to these random variables:

$$Y_1 = x + E_1 \ \ and \ \ Y_2 = x + E_2 \tag{2.1}$$

where E is the so-called "observational" noise. Note in (2.1) x is considered as an unknown but *deterministic* quantity. Notice that we didn't consider natural variability of x in this simple example, and assuming that we are concerned only with the measurement error of

[1] In the following, upper case letters X, Y, E... will represent random variables from which lower case letters x, y, ϵ... will represent some realizations.

Preceding page blank

x (for a more general formulation, see Section 2.2). Observations Y are, however, random variables through the observational random noise process E.

With this formalism, observations y_1 and y_2 are particular realizations of the two random variables Y_1 and Y_2:

$$y_1 = x + \epsilon_1 \;\; and \;\; y_2 = x + \epsilon_2 \tag{2.2}$$

where ϵ_1 and ϵ_2 are realizations of the random variables E_1 and E_2, respectively.

A common way to deal with random processes is to examine their statistical moments after application of the statistical mean operator $E\{\}$. The statistical mean can be seen as an average of all possible states, (Jazwinsky, 1970). Since we assume that the instruments are unbiased, the statistical means of E_1 and E_2 are null, i.e.,

$$E\{E_1\} = 0 \quad \text{and} \quad E\{E_2\} = 0 \tag{2.3}$$

The second moments, also called dispersion or variance, of the random variables E_1 and E_2 are directly related to uncertainty. More precisely we have:

$$E\{E_1^2\} = \sigma_1^2 \quad \text{and} \quad E\{E_2^2\} = \sigma_2^2 \tag{2.4}$$

Also, we suppose that the instruments are completely independent, so that there is no correlation between the observational noises:

$$E\{E_1 E_2\} = 0 \tag{2.5}$$

The estimator X^* is also a random variable and is sought as a linear combination of the two measurements:

$$X^* = a_1 Y_1 + a_2 Y_2 \tag{2.6}$$

where the weights a_1 and a_2 are to be determined. We first want the estimator to be unbiased, thus $E\{X^*\} = x$. Since a_1 and a_2 are deterministic and using relation (2.1) and (2.6), we can calculate the statistical mean of this estimator:

$$\begin{aligned} E\{X^*\} = x &= E\{a_1 Y_1 + a_2 Y_2\} \\ &= a_1 E\{Y_1\} + a_2 E\{Y_2\} \\ &= a_1 x + a_2 x \end{aligned} \tag{2.7}$$

Hence:

$$a_1 + a_2 = 1 \tag{2.8}$$

Among all the unbiased linear estimators which satisfy (2.8), we select the estimator which minimizes the dispersion of the estimation around the true value. This dispersion can be seen as the estimation error and is given by the variance of the estimator:

$$\sigma^2 = E\{(X^* - E\{X^*\})(X^* - E\{X^*\})\} \tag{2.9}$$

Using equations (2.5), (2.6) and the fact that $E\{X^*\} = x$ we get from (2.9):

$$\begin{aligned}
\sigma^2 &= E\{(X^* - E\{X^*\})(X^* - E\{X^*\})\} \\
&= E\{(a_1 Y_1 + a_2 Y_2 - x)^2\} \\
&= E\{(a_1(Y_1 - x) + a_2(Y_2 - x) + (a_1 + a_2 - 1)x)^2\} \\
&= E\{(a_1(Y_1 - x) + a_2(Y_2 - x))^2\} \\
&= a_1^2\sigma_1^2 + a_2^2\sigma_2^2
\end{aligned} \tag{2.10}$$

Using equations (2.8), we can easily find the minimum of the function $\sigma^2(a_1, a_2)$. The corresponding coefficients are:

$$a_1 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \quad \mathit{and} \quad a_2 = \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \tag{2.11}$$

which gives the following estimator:

$$X^* = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}\, Y_1 \;+\; \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}\, Y_2 \tag{2.12}$$

The estimate x^*, which is deterministic, is a realization of the random variable X^* with this set of coefficients:

$$x^* = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}\, y_1 \;+\; \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}\, y_2 \tag{2.13}$$

where y_1 and y_2 are the actual measured values.

One can see that, if, for instance, the first observation is very poorly known or even absent ($\sigma_1 \to \infty$), then the estimate is equal to the second measurement $x^* = y_2$. On the contrary, if this observation tends to be perfect ($\sigma_1 \to 0$), then the estimate is equal to this value $x^* = y_1$.

Using equations (2.9) and (2.10), we can calculate the variance of the estimator. This variance represents the estimation error and is given by:

$$\sigma^2 = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2} \tag{2.14}$$

An examination of the estimation error in equation (2.14) shows that $\sigma^2 \leq \sigma_1^2$ and $\sigma^2 \leq \sigma_2^2$. Thus, the uncertainty is decreased in the combination of the two measurements. In addition, equation (2.14) can be rewritten as:

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} \tag{2.15}$$

which has a simple interpretation: if one calls "precision" the inverse of the dispersion or variance, then the precision of the estimate is the sum of the precisions of the observations. Because it minimizes the estimation error, the estimator presented above is usually called the best linear unbiased estimator (BLUE).

The same estimate x^* can be found through a different approach: an acceptable estimate of the exact value must be close to the observations, at least within the accuracy of the latter. For any value x, the distance between x and the observations can be measured by the following quadratic quantity:

$$J(x) = \frac{(x - y_1)^2}{\sigma_1^2} + \frac{(x - y_2)^2}{\sigma_2^2} \tag{2.16}$$

Because the function J penalizes the deviation between the truth and the observations, it is usually called *cost function.* In such a formulation, observations are taken into account with weights corresponding to their precision, so that a better estimation can be expected. The estimate x^* is the value which minimizes the cost function. At its minimum x^* the derivative of J is null, so that:

$$\frac{dJ}{dx}(x^*) = 0 = 2\frac{(x^* - y_1)}{\sigma_1^2} + 2\frac{(x^* - y_2)}{\sigma_2^2} \tag{2.17}$$

hence:

$$x^* = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}\, y_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}\, y_2 \tag{2.18}$$

which is similar to (2.13).

This variational formulation of the estimation problem is conceptually very simple and requires fewer calculations than the full probabilistic solution. However, there is no counterpart of equation (2.14): the variational formalism does not give any indications of the error associated with the solution. Here, the adjoint equation does not appear explicitly because our example is monodimensional.

2.2 Data assimilation

2.2.1 Optimal statistical estimation

Section 2.1 illustrated some of the basic concepts of the linear estimation theory through a simple example and showed how observational information can be best processed so as to reduce uncertainty in estimates. Note that, in the previous simple example, the variable x was assumed to be unknown but certain. A more realistic approach would be to consider X as a random variable. This is particularly necessarily in atmospheric studies in which X represents some physical phenomenon that could never be perfectly known and there will always be some unpredictable fluctuations that can only be represented by stochastic variables. In addition, in practice, atmospheric studies are generally conducted on a geographical domain and not at a single location and several parameters at different locations are necessary to properly describe the state of the studied physical field over this domain. Consequently, the quantity to estimate X is a random vector of dimension N and components $(X^1, ..., X^N)$. For instance, X can be thought of as the values of the temperature at the points of a two- or a three-dimensional grid mesh. Similarly, observations are not restrained to a single location, but they usually come from an observational network. Thus, the observation Y is also a vector of dimension M, not necessarily equal to N, of components $(Y^1, ..., Y^M)$. In addition, these observations are rarely recorded on a regular grid, so that, generally, the grid on which the observations are availaible differs from the grid on which the estimated field is sought. In order to get a relationship between the estimated field and the observations equivalent to (2.1), we have to interpolate X to the observation locations. H is the operator performing this interpolation; then in the vectorial case the relationship corresponding to equation (2.1) case is:

$$Y = HX + E \tag{2.19}$$

Where X, Y and E are random vectors, *i.e* vectors made of random variables. The observation operator H is an MxN matrix. Note that, in the formalism (2.19) the observations could be different from the field to be estimated. In such a case H would also represent some physical transformations, for instance the calculation of satellite radiance from pressure, temperature and humidity profiles.

As before the observational noise is assumed to have zero mean, i.e., its mean is a null vector:

$$E\{ E \} = [E\{E^1\}, ..., E\{E^M\}]^T = [0, ..., 0]^T \tag{2.20}$$

where the superscript $(\ )^T$ denotes the transposition operation. To describe the second order statistical properties of the observational noise, we now introduce the so-called *variance-covariance* matrix:

$$O = E\{ (E - E\{E\})(E - E\{E\})^T \} = E\{ EE^T \} \tag{2.21}$$

which can be explicitly written as

$$O = \begin{pmatrix} E\{E^1E^1\} & ... & E\{E^1E^i\} & ... & E\{E^1E^M\} \\ ... & ... & ... & ... & ... \\ E\{E^iE^1\} & ... & E\{E^iE^i\} & ... & E\{E^iE^M\} \\ ... & ... & ... & ... & ... \\ E\{E^ME^1\} & ... & E\{E^ME^i\} & ... & E\{E^ME^M)\} \end{pmatrix} \tag{2.22}$$

where O is an $M \times M$ matrix whose diagonal elements are the variances of observations while the off-diagonal elements are the spatial covariances between various observations. It shall also include "representative" error. This matrix is assumed to be known.

Because the dimension of observation vector (M) is usually not the same as the dimension of the vector to estimate (N), observations are often insufficient to completely determine all the components, i.e., $M < N$. In this case (as we normally face for the atmospheric and oceanic data assimilation problems), the estimation problem cannot be solved in the absence of other sources of information. In numerical prediction, additional information is usually available in the form of a forecast that results from the previous analysis cycle. This means that an estimate x_b, which does not take into account the new observations, is also available. The estimation problem will consist of updating this old estimate with the new observational information. Here we touch on the very important concepts of *a priori* and *a posteriori* information. These concepts are of fundamental importance. We will, therefore, assume that, in addition to Y and O, a realization x_b of the N dimensional random vector estimator X_B is also available with its statistics B.

(The letter B stands for *background* information). Mathematically, this adds the following relationships (the estimator is assumed to be unbiased for simplicity):

$$E\{\ X_B\ \} = E\{\ X\ \}\ = \bar{x}\ \text{ and }\ B = \{\ (X_B - X)(X_B - X)^T\ \} \tag{2.23}$$

These statistics characterize the *a priori* probability distribution of the background information. One can now easily anticipate that the key information in this estimation problem will be the *a posteriori* probability, *i.e* the probability distribution of the random variable X after incorporation of the observation information Y.

In summary, the quantities assumed known in this estimation problem are the observation operator H, a realization y with its uncertainty matrix O of the observations random vector Y and a *prior* estimate x_b (a realization of the random variable X_B) with its uncertainty matrix B.

In order to estimate X, we now proceed as in section 2.1, and find an estimator X^* which is a linear function of the available information,

$$X^*\ = A_1 X_B + A_2 Y \tag{2.24}$$

Now, A_1 is a $N \times N$ matrix and A_2 is $N \times M$ matrix. Both A_1 and A_2 are to be determined. Again, we want the estimator to be unbiased, *i.e* $E\ \{\ X^*\} = \bar{x}$. This condition is verified if:

$$\bar{x}\ =\ E\{\ X^*\}\ =\ E\{\ A_1 X_B + A_2 Y\ \}\ =\ A_1\bar{x} + A_2 H\bar{x} \tag{2.25}$$

Hence, if:

$$A_1\ =\ I_N - A_2 H \tag{2.26}$$

where I_N is the unit matrix of order N. Equation (2.26) allows us to rewrite the estimator (2.24) as the background variable plus a correction term:

$$X^*\ =\ X_B\ +\ A_2(Y - HX_B) \tag{2.27}$$

The correction term $(Y - HX_B)$ is also called the *innovation* vector. This vector represents the difference between the new information brought in by the observations Y and what is already known from the prior information X_B.

In addition, among all the matrices satisfying (2.26), we want to choose the one which minimizes the estimation error. This error is the vector $\tilde{X}$ made of the elementary errors associated with the estimation of each component of the vector X:

$$\tilde{X} = X^* - X = [\, X_1^* - X_1, ..., X_N^* - X_N \,] \tag{2.28}$$

To quantify this error, we can calculate the norm of the error vector after, eventually, averaging on all the possible states:

$$\sigma = |\tilde{X}| = \left(\sum_{i=1}^{i=N} E\{\, \tilde{X}^i \tilde{X}^i \} \right)^{1/2} \tag{2.29}$$

To calculate this norm, we can use the error covariance matrix $P = E\{\, \tilde{X}\tilde{X}^T \,\}$. associated with the estimator:

$$P = \begin{pmatrix} E\{\tilde{X}^1\tilde{X}^1\} & ... & E\{\tilde{X}^1\tilde{X}^i)\} & ... & E\{\tilde{X}^1\tilde{X}^N\} \\ ... & ... & ... & ... & ... \\ E\{\tilde{X}^i\tilde{X}^1\} & ... & E\{\tilde{X}^i\tilde{X}^i)\} & ... & E\{\tilde{X}^i\tilde{X}^N\} \\ ... & ... & ... & ... & ... \\ E\{\tilde{X}^N\tilde{X}^1\} & ... & E\{\tilde{X}^N\tilde{X}^i)\} & ... & E\{\tilde{X}^N\tilde{X}^N\} \end{pmatrix} \tag{2.30}$$

As we can see, the norm of the estimation error vector is equal to the square root of the sum of the diagonal terms or *trace* of P. The variance-covariance matrix contains almost all the information necessary to solve the problem. If we remember that the trace operator is a scalar product for the matrices, then *tr*(P) is the norm of P. We can see that the terms of P describe the spatial structures of the error, while its norm provides a quantification of this error.

Let us express the covariance matrix P in a function of the unknowns X and A_2. Using (2.19) and (2.26), we can develop the estimation error vector:

$$\begin{aligned} \tilde{X} &= X' - X \\ &= A_1 X_B + A_2 Y - X \\ &= A_1(X_B - X) + A_2(Y - HX) + (A_1 + A_2 H - I_N)X \end{aligned} \tag{2.31}$$

But according to equation (2.26) the third term in the right-hand-side of the last line is null. Again using (2.26), we can write this error in a simpler way:

$$\tilde{X} = (I_N - A_2 H)(X_B - X) + A_2(Y - HX) \tag{2.32}$$

With this expression of the estimation error, the covariance matrix P is:

$$\begin{aligned}
P &= E\{ \tilde{X}\tilde{X}^T \} \\
&= E\{ ((I_N - A_2H)(X_B - X) + A_2(Y - HX))\,((I_N - A_2H)(X_B - X) + A_2(Y - HX))^T \} \\
&= \quad (I_N - A_2H)E\{ (X_B - X)(X_B - X)^T \}(I_N - A_2H)^T \\
&\quad + (I_N - A_2H)E\{ (X_B - X)(Y - HX)^T A_2^T \} \\
&\quad + A_2E\{ (Y - HX)(X_B - X)^T \}(I_N - A_2H)^T \\
&\quad + A_2E\{ (Y - HX)(Y - HX)^T A_2^T \}
\end{aligned} \tag{2.33}$$

In addition we have:

$$\begin{aligned}
E\{ (X_B - X)(X_B - X)^T \} &= B \\
E\{ (Y - HX)(Y - HX)^T \} &= E\{ EE^T \} = O \\
E\{ (X_B - X)(Y - HX)^T \} &= E\{ (X_B - X)E^T \} = 0 \\
E\{ (Y - HX)(X_B - X)^T \} &= E\{ E(X_B - X)^T \} = 0
\end{aligned} \tag{2.34}$$

The last two lines simply state that the new measurements are not correlated with the past estimation. Thus:

$$P = (I_N - A_2H)\, B\, (I_N - A_2H)^T + A_2\, O\, A_2^T \tag{2.35}$$

The norm of the estimation error associated with our estimator X^* will be a minimum if we can find the matrix A_1 and A_2 satisfying (2.26) and minimizing the trace of P. This minimization can be easily done using the following theorems on trace:

$$\begin{aligned}
trace(\, A + B \,) &= trace(\, A \,) + trace(\, B \,) \\
\frac{\partial}{\partial A} trace\,(\, ABA^T \,) &= 2AB
\end{aligned} \tag{2.36}$$

for any symmetric matrix B (Gelb, 1980). Equation (2.36) can be easily demonstrated using the properties of the scalar product. Since covariances matrices are symmetric, we have:

$$\begin{aligned}
\frac{\partial}{\partial A_2} trace\,(\, P \,) &= \frac{\partial}{\partial A_2} trace\,(\, (I_N - A_2H)B(I_N - A_2H)^T \,) + \frac{\partial}{\partial A_2} trace\,(\, A_2OA_2^T \,) \\
&= -2(I_N - A_2H)BH^T + 2A_2O
\end{aligned} \tag{2.37}$$

At the minimum, this derivative should be 0. Thus the matrix A_2 satisfies

$$0 = -2(I_N - A_2H)BH^T + 2A_2O \tag{2.38a}$$

or:

$$0 = -BH^T + A_2[\, HBH^T + O\,] \tag{2.38b}$$

Solving for A_2, we obtain the optimal value:

$$A_2 = BH^T[\, HBH^T + O\,]^{-1} \tag{2.39}$$

A_2 is called the *Kalman Gain.*

Substituting (2.39) into (2.27) gives an expression for the unbiased minimum variance estimator:

$$X^* = X_B + BH^T[\, HBH^T + O\,]^{-1}\,(Y - HX_B) \tag{2.40}$$

This estimator is the so-called *Kalman filter.*

Similarly, the expression of the error covariance matrix associated with this estimator can be obtained by multiplying (2.38a) by A_2^T, which gives:

$$\begin{aligned} 0 &= -2(I_N - A_2H)BH^TA_2^T + 2A_2OA_2^T \\ 0 &= (I_N - A_2H)B(I_N - A_2H)^T + A_2OA_2^T - (I_N - A_2H)B \\ 0 &= P - (I_N - A_2H)B \end{aligned} \tag{2.41}$$

Where we have made use of (2.35). Therefore, the covariance matrix is:

$$\begin{aligned} P &= (I_N - A_2H)B \\ &= B - BH^T[\, HBH^T + O\,]^{-1}HB \end{aligned} \tag{2.42}$$

Since B and O are both symmetric positive definite, $BH^T[\, HBH^T + O\,]^{-1}HB$ is also symmetric positive and therefore [3] $P \leq B$, *i.e.*, the uncertainty has decreased during the estimation.

Finally, the estimate is a realization of the estimator X^*. This estimate is unbiased and minimizes the estimation error. This error is given by the covariance matrix P:

$$x^* = x_b + BH^T[\, HBH^T + O\,]^{-1}\,(y - Hx_b) \tag{2.43}$$

[3] $P \leq B$ if $p_{i,j} \leq b_{i,j} \ \forall\, i,\, j$

$$P = B - BH^T[\, HBH^T + O\,]^{-1}HB \tag{2.44}$$

The statistical solution (2.43) has been the basis of most of the operational analysis techniques and some are still in application. This analysis procedure is called *Optimal Interpolation* (OI) and is most usually implemented in the following way (Talagrand 1992): A numerical meteorological model is used to integrate the results of the previous analysis to the time of the new analysis. This provides the background term x_b; while all the observations that are made available during the corresponding analysis time period are collected to build the vector y. At the end of the model integration, the statistical estimate x^* is computed using the expression (2.43). This implies that the matrices O and B are known. O is characteristic of the instrumental noise. This noise can be studied in laboratory, so that O or, at least its diagonal, is usually known with good precision. The B matrix is much harder to evaluate. There is a rigorous way to propagate in time the information contained in B using the meteorological model, but the implementation of such a solution requires computational resources that far exceed the present computer's capacity. B has to be approximated; it is usually modeled on the basis of a number of simple hypotheses on the shape and spatial decay of the corresponding covariance functions.

Computer limitations are also the cause of another kind of approximation called *data selection* (Talagrand, 1992). In the *data selection* algorithm, the analysis grid is divided into subsets and the computation of (2.43) is repeatedly applied to these different subsets. However, for each subset, a limited number of observations, namely the observations located in the vicinity of the subset, are used. Thus, the size of the matrices that must be inverted is considerably reduced. This selection of observations is certainly legitimate in the sense that observations at a large distance from a given point must have a small influence on the value of the estimated field at that point. However experience shows that it nevertheless introduces spatial noise in the resulting field. Equation (2.44) is usually not fully implemented, and only the diagonal terms of the matrix P, *i.e.*, the variances of the analysis error, are computed.

2.2.2 Variational Approach — 3D-VAR

As in the example in Section 2.1, we ask if there is a deterministic formulation of the multidimensional estimation problem. The answer is yes and this deterministic formulation is also variational. As in the unidimensional case, the variational formulation is based on the concept of precision instead of dispersion as with probability. Precision corresponds to the inverse of the variance-covariance matrix. Therefore, the variational statistical formulation of the multidimensional estimation problem can be expressed by the following cost function:

$$J_{VAR} = 1/2(x_b - x)^T B^{-1}(x_b - x) + 1/2(y - Hx)^T O^{-1}(y - Hx) \tag{2.45}$$

The vector x' which minimizes J_{VAR} can be interpreted as the values that best fit simultaneously the background information x_b and the observations y, given their respective degree of confidence or precision B^{-1} and O^{-1}.

To find the minimum of J_{VAR}, we can calculate its derivative:

$$\frac{\partial J}{\partial x} = - B^{-1}(x_b - x) - H^T O^{-1}(y - Hx) \tag{2.46}$$

At the minimum this derivative is null, thus:

$$0 = - B^{-1}(x_b - x') - H^T O^{-1}(y - Hx') \tag{2.47}$$

which can be solved for x':

$$0 = B^{-1}(x_b - x') + H^T O^{-1}(y - Hx_b + H(x_b - x')) \tag{2.48}$$

which gives:

$$x' = x_b + [\, B^{-1} + H^T O^{-1} H \,]^{-1} H^T O^{-1}(y - Hx_b) \,] \tag{2.49}$$

However, a null derivative only indicates an extremum, which can be a minimum or a maximum. Let's check that, in addition to a null derivative, the second derivative or *Hessian matrix* is positive:

$$\begin{aligned} \frac{\partial^2 J}{\partial x^2} &= \frac{\partial}{\partial x}\left(- B^{-1}(x_b - x) - H^T O^{-1}(y - Hx) \right) \\ &= B^{-1} + H^T O^{-1} H \end{aligned} \tag{2.50}$$

which characterizes a minimum. Since B and O are symmetric positive definite, the matrix $[B^{-1} + H^T O^{-1} H]$ is also a positive definite and the extremum x' given by (2.49) is a minimum.

Comparing (2.49) and (2.43), it is not readily apparent that the variational solution and the minimal variance estimate are equivalent. This equivalence is shown in Appendix A.

Note, that the variational solution does not provide an estimation of the error associated with its solution. However, if we look at the expression of the Hessian matrix in (2.50) we can calculate its inverse using the Woodbury formula[4] for $\Lambda = B^{-1}$, $\Gamma = H^T$ and $\Theta = O^{-1}H$, this gives:

$$\begin{aligned}\left(\frac{\partial^2 J}{\partial x^2}\right)^{-1} &= [\,B^{-1} + H^T O^{-1} H]^{-1} \\ &= B - BH[\,I + O^{-1}HBH^T\,]^{-1} O^{-1} HB \\ &= B - BH\left[\,O[\,I + O^{-1}HBH^T\,]\,\right]^{-1} HB \\ &= B - BH[\,O + HBH^T\,]^{-1} HB\end{aligned} \tag{2.51}$$

which is the expression (2.44) of the covariance matrix P associated with the estimation solution.

We have, thus, the following fundamental result:

$$P = \left(\frac{\partial^2 J}{\partial x^2}\right)^{-1} \tag{2.52}$$

That is, the error associated with the variational solution is given by the inverse of the Hessian matrix. Using (2.44) and (2.51), we can now express the inverse P^{-1} of the covariance matrix:

$$P^{-1} = \left(\frac{\partial^2 J}{\partial x^2}\right) = B^{-1} + H^T O^{-1} H \tag{2.53}$$

which allows an interpretation in term of precision as in the unidimensional case. The precision or information that accompanies the minimal variance estimate is equal to the sum of the precision on the background information and the precision on the observations passed through the observation operator.

4

$$[\,\Lambda + \Gamma\Theta\,]^{-1} = \Lambda^{-1} - \Lambda^{-1}\Gamma(\,I + \Theta\Lambda^{-1}\Gamma\,]\Theta\Lambda^{-1}$$

Data selection mentioned in Section 2.2.1 tends to be used less and less with the emergence of iterative techniques. Instead of attempting to invert directly the matrix $[O+HBH^T]$, which represents the main computational burden in solving (2.43), a series of approximate solutions are found by an iterative procedure that converges toward the exact solution. Such implementations are referred as *3 Dimensional Variational Assimilation* (3D-VAR) system.

2.2.3 Physical space analysis system — 3D-PSAS

The success of the iterative techniques introduced by the variational formulation of the 3D data assimilation problem has produced an interesting feedback effect on the OI procedure. Iterative solutions, like Conjugate Gradient descent, have been used for a long time in numerical analysis to solve matrix equations. They can, therefore, be used to invert the matrix $[O+HBH^T]$ in (2.43). Such an approach provides a tractable implementation of the complete OI solution. Paradoxically, in this kind of implementation the statistical solution:

$$x^* = x_b + BH^T[O+HBH^T]^{-1}(y - Hx_b) \tag{2.54}$$

is formulated as the minimum of a new cost function:

$$J_{PSAS} = 1/2\, (w - w_B)^T[O+HBH^T](w - w_B) - (w - w_B)^T(y - Hx_b) \tag{2.55}$$

where $w = (BH^T)^{-1}x$ and $w_B = (BH^T)^{-1}x_b$. As we can see, the new variable w is the counterpart of x in the physical space of the observations. For that reason, this implementation is called *3-Dimensional Physical Space Analysis System* (3D-PSAS).

The cost function J_{PSAS} is primarily introduced for practical reasons. It simply makes easier the implementation of the iterative algorithm used to invert the matrix $[O+HBH^T]$. Although J_{PSAS} seems quite different from the cost function J_{VAR}, both functions are equivalent; they express the same concept in two different spaces: the variational solution is expressed in the phase space while the 3D PSAS uses the physical space for its operations. One expression can be deduced from the other by the transformations performed in (A1-A7) in Appendix A. The 3D-VAR and 3D-PSAS differ in required approximation to B. In 3D-VAR, B^{-1} is required and in 3D-PSAS B is required instead.

In summary, 3D-VAR and 3D-PSAS iterative algorithms are the two most efficient implementations of the statistical solution for the data assimilation problem. In essence, both algorithms are equivalent and it was shown (Courtier 1997) that their overall computing cost is similar.

2.2.4 4-Dimensional variational data assimilation — 4D-VAR

Up to now, we have always assumed that the observations were available at the time when the estimation is performed, and analyses are performed at exact intervals depending on their update cycle (every 6, 12, 24 hours). With such approximation, it is not possible to take into account the temporal variability of the observations within the analysis cycle. Depending on the type of observation, this can result in an important loss of information, particularly for high frequency observations such as satellite and radar data. In reality, observations are issued from different observational networks, each network having its own measurement frequency.

In OI, 3D-VAR, and 3D-PSAS, a meteorological numerical model is used to propagate the background information following an analysis cycle. This numerical model is another source of information that is used in 4-dimensional data assimilation. This new information is necessary to balance the increase of degrees of freedom in the estimation problem that results from the extension to the temporal dimension. The numerical model which usually discretizes the fluid dynamic equations can be considered as *a prior* information. Its equations are a convenient medium to encapsulate all the statistical, dynamical and physical knowledge we *a priori* have on the atmosphere. This model is supposed to reflect the evolution of the vector to estimate X (the truth) with a certain degree of uncertainty and has usually the form of a differential equation:

$$\frac{\partial X(t)}{\partial t} = F(\, X(t)\,) + W(t) \tag{2.58}$$

where F stands for all the mathematical functions involved in the meteorological model, and $W(t)$ is a random variable figuring the model error. As usual, $W(t)$ is assumed to have a 0 mean and a covariance matrix $Q(t)$. In addition we assume that $W(t)$ is a white process, *i.e* $E\{W(t), W^T(t')\} = 0$ if $t' \neq t$.

The information available in the 4D data assimilation problem is, therefore,: i) a background term x_b with its covariance matrix B, generally available at the beginning of the assimilation time period; ii) the meteorological model of (2.58) and iii) the observations which are now distributed in time;

$$Y(t) \;=\; H(\,X(t)\,) \;+\; E(t) \tag{2.59}$$

Where H figures the configuration of the observational network at time t and $E(t)$ is random variables accounting for the measurement error. As for the model, the error $E(t)$ is assumed to have a 0 mean and a covariance matrix $O(t)$. $E(t)$ is also assumed to be white and uncorrelated with the model error $W(t)$, *i.e.* $E\{\ EW^T\ \} = 0$.

From these data, two different approaches are possible:

- The filtering solution is sequential and aims to find the best estimate at the time the observations are available. This is the approach used in OI and best illustrated by the *Kalman filter* algorithm in which the optimal 3D data assimilation procedure described in section 2.2.1 is applied each time an observation becomes available.

- The smoothing solution aims to globally estimate the state $X(t)$ on a complete time period $[t_0, t_R]$ using all the observations available during this time period. In this global adjustment, it makes use of the numerical model to take into account the temporal distribution of the observations.

From a probability viewpoint, these solutions differ from the *posterior* information they use: the filtering solution considers the information contained in the conditional probability function $p(X(t)/Y(\tau),\ t_0 \leq \tau \leq t)$, while the smoothing solution is based on the conditional probability function $p(X(t)/Y(\tau),\ t_0 \leq \tau \leq t_R)$, *i.e.* by using all the information available before, but also after, the estimation time. Thus, the smoothing can only be performed at the end of the assimilation time window; but, except for the value $X(t_R)$ at the end of the assimilation time period, the smoothing solution processes much more information than the filter does. Better estimates should, therefore, be expected from the smoothing solution.

Indeed, the *Kalman smoother* equations which provide the optimal solution (in the sense that it minimizes the variance) show that: i) the smoothing estimation error is lower than the filtering estimation error and ii) both solutions are identical, same estimate and same covariance matrix, at the end of the estimation period. In fact filtering can be viewed as a pre-processing operation of the smoothing process. We now show the 4-D data assimilation problem using a variational formulation. For notation purposes, we will use the continuous formulation which leads to more compact expressions. In reality, observations are available at discrete times, but this does not affect the general form of the solution which can be easily obtained from the continuous formulation using a Dirac distribution.

Now, the cost function must account for all the observations distributed within the assimilation time window $[t_0,\ t_R]$. A natural extension of the 3D cost function (2.45) to the temporal dimension could be (we note $x_{t_0} = x(t_0)$, , $x_t = x(t)$ and $\dot{x} = dx/dt$):

$$J \ = \ 1/2(x_{t_0} - x_b)^T B^{-1}(x_{t_0} - x_b) \ + \ 1/2 \int_{t_0}^{t_R} (y_t - H(x_t))^T O_t^{-1}(y_t - H(x_t)) \ dt \quad (2.60)$$

Note that this cost function can also handle the 3D case with $t_R = t_0$ and $O(t) \ = \ O * \delta(t-t_0)$. However, this function is not complete in the sense that it does not reflect all the information contained in the problem. Indeed, the meteorological model is the particular information not used in this formulation. The model information is somewhat apart, because it does not provide an observational fact, but information on the shape of the phenomenon to estimate. Since, we know that the truth X satisfies (2.58) with a given uncertainty W, it should be the same for the estimate; otherwise, this estimate will inevitably be biased. So, we are no more interested in the absolute minimum of the cost function J, but in the model equations solution x^* leading to the smallest value of J. In this prospect, the model error W appears as an additional term which should be sought to be minimized. Consequently, an additional term has to be added to the simple cost function (2.60):

$$\begin{aligned} J &= J \ + \ 1/2 \int_{t_0}^{t_R} (\dot{x} - F(x_t))^T Q_t^{-1}(\dot{x} - F(x_t)) \\ &= J \ + \ 1/2 \int_{t_0}^{t_R} w_t^T Q_t^{-1} w_t \ dt \end{aligned} \quad (2.61)$$

and this function has to be minimized under constraint of the model equations (2.58).

The minimization problem of 4D data assimilation slightly differs from its 3D counterpart which was simply an unconstrained optimization problem. The 4D constraint minimization can be reduced to an unconstrained optimization problem by considering the minimization of the *Augmented Lagrangian* function of J, instead of J itself. The Lagrangian function L is derived from J by the following expression:

$$L \ = \ J \ + \ \int_{t_0}^{t_R} \lambda_t^T(\dot{x} - F(x_t) - w_t) \ dt \quad (2.62)$$

where λ_t is a N dimensional vector to be defined. Using the calculus of variation it can be shown (Courant and Hilbert 1989) that J and L have the same extrema. These extrema are solutions of the so-called *Euler Lagrange* equations which express the stationarity for L with respect to all its input variables: $x_{t_0}, \dot{x}_{t_0}, x_t, \dot{x}_t, x_{t_R}, \dot{x}_{t_R}, w_t, \dot{w}_t, \lambda_t$ and $\dot{\lambda}_t$ which should be considered as independent. The solution to the problem of minimizing J in (2.61) or L in (2.62) can be found by introducing an adjoint model (see Section 2.3) and a standard unconstrained minimization algorithm (Section 3.8).

2.3 Introduce an adjoint model

2.3.1 A continuous form of adjoint model

As mentioned in section 2.2.4, the cost function J in (2.61) and its *Augmented Lagrangian* function L in (2.62) have the same extrema. These extrema are solutions of the so-called *Euler Lagrange* equations (Courant and Hilbert 1989). In order to get an expression of the Euler-Lagrange equations, we write L explicitly:

$$\begin{aligned} L = \quad & 1/2\ (x_{t_0} - x_b)^T B^{-1} (x_{t_0} - x_b) \\ + \ & 1/2 \int_{t_0}^{t_R} (y_t - H(x_t))^T O_t^{-1} (y_t - H(x_t)) + \ 1/2 \int_{t_0}^{t_R} w_t^T Q_t^{-1} w_t \\ + \ & \int_{t_0}^{t_R} \lambda_t^T (\dot{x} - F(x_t) - w_t)\ dt \end{aligned} \tag{2.63}$$

An integration by parts of the last (non quadratic) term in the right-hand side of (2.63) gives:

$$\begin{aligned} \int_{t_0}^{t_R} \lambda_t^T (\dot{x} - F(x_t) - w_t)\ dt &= \int_{t_0}^{t_R} \lambda_t^T \dot{x} dt - \int_{t_0}^{t_R} \lambda_t^T (F(x_t) + w_t)\ dt \\ &= [\lambda_t x_t\ ]_{t_0}^{t_R} - \int_{t_0}^{t_R} \dot{\lambda}_t^T x_t dt - \int_{t_0}^{t_R} \lambda_t^T (F(x_t) + w_t)\ dt \end{aligned} \tag{2.64}$$

and thus:

$$\begin{aligned}
L = \; & 1/2\;(x_{t_0}-x_b)^T B^{-1}(x_{t_0}-x_b) \\
& + 1/2 \int_{t_0}^{t_R} (y_t - H(x_t))^T O_t^{-1}(y_t - H(x_t)) + 1/2 \int_{t_0}^{t_R} w_t^T Q_t^{-1} w_t \\
& + \lambda_{t_R} x_{t_R} - \lambda_{t_0} x_{t_0} - \int_{t_0}^{t_R} \dot{\lambda}_t^T x_t dt - \int_{t_0}^{t_R} \lambda_t^T (F(x_t) + w_t)\, dt \\
= J & + \lambda_{t_R} x_{t_R} - \lambda_{t_0} x_{t_0} - \int_{t_0}^{t_R} \dot{\lambda}_t^T x_t dt - \int_{t_0}^{t_R} \lambda_t^T (F(x_t) + w_t)\, dt
\end{aligned} \qquad (2.65)$$

With this expression of L, we can now perform the derivatives of L with respect to all the input variables and find the expression of the Euler-Lagrange equations for the continuous 4-D-VAR data assimilation problem:

$$\lambda_{t_0} - B^{-1}(x_{t_0}^* - x_b) = 0 \qquad (2.66)$$

$$\lambda_{t_R} = 0 \qquad (2.67)$$

$$-\frac{\partial \lambda}{\partial t} - \frac{\partial F}{\partial x}^T \lambda_t - \frac{\partial H}{\partial x}^T O_t^{-1}(y_t - H(x_t^*)) = 0 \qquad (2.68)$$

$$\lambda_t - Q_t^{-1} w_t = 0 \qquad (2.69)$$

$$\frac{\partial x^*}{\partial t} - F(x_t^*) - w_t = 0 \qquad (2.70)$$

This constitutes the so-called optimality system of equations that the estimate x^* should satisfy to be the optimal solution of the 4D-VAR data assimilation problem. Equation (2.70) is precisely the model equation (2.58), this ensures that the model constraint will be enforced for the optimal solution.

Using (2.69), we can eliminate w_t in (2.70), this gives the following system of coupled equations:

$$x_{t_0}^* = x_b + B\lambda_{t_0} \qquad (2.71)$$

$$\frac{\partial x^*}{\partial t} = F(x_t^*) + Q(t)\lambda_t \qquad (2.72)$$

$$-\frac{\partial \lambda}{\partial t} = \frac{\partial F}{\partial x}^T \lambda_t + \frac{\partial H}{\partial x}^T O_t^{-1}(y_t - H(x_t^*)) \qquad (2.73a)$$

$$\lambda_{t_R} = 0 \qquad (2.73b)$$

In particular, we see that λ is the solution of a partial differential equation (2.73) similar to the model equation (2.72), except that the model physics F has been linearized and transposed. For this reason, (2.73) is called the *adjoint model* and λ is called the *adjoint state* associated to state x. In the following we will refer to the meteorological model as the *direct* model when confusions with its adjoint counterpart are possible.

Note an interesting relation which will be used in 4D-VAR algorithm: At the stationary point, $\frac{\partial L}{\partial x_0} = 0$, which is applied to the last expression of L in (2.65), leads to:

$$\frac{\partial J}{\partial x_0} - \lambda_{t_0} = 0 \tag{2.74}$$

In other words, the value of the adjoint state λ_{t_0} at the initial time is equal to the value of the gradient of the cost function J with respect to the initial conditions (ICs) x_{t_0}.

Here, the adjoint model was introduced for solving a minimization problem. We note the minus sign in front of the time derivative of the adjoint state in (2.73) which indicates that this equation should be integrated backward in time from t_R to t_0, with the appropriate "initial" conditions given by (2.73): $\lambda_{t_R} = 0$. There is, thus, no information initially introduced in the adjoint model, but information is only provided by the forcing term $(\frac{\partial H}{\partial x}^T O_t^{-1}(y_t - H(x_t^*)))$ which expresses the deviation of the model prediction $(H(x_t))$ from the corresponding observations ($\mathbf{y}$). Note that, this term has a "precision" aspect (matrix O^{-1}). The adjoint model can, therefore, be interpreted as a computational operator which propagates backward the gain of information that results from the observations. The total gain is contained in the vector λ_{t_0} which takes into account all the observations available during the assimilation period $[t_0, t_R]$. In (2.71), this gain of information is used to correct the outdated value x_b of the background information. Similarly, the error term $Q\lambda_t$ is added in (2.71) to account for the information brought by the observations that was not already present in the physics of the model (operator F). The coupling in (2.72) and (2.73) shows how observational and *a priori* information is intricate.

The 4D-VAR data assimilation algorithm is obtained when the model is assumed to be perfect. This means that $E\{W\} = 0$ and $Q = E\{WW^T\} = 0$, *i.e.* there is no dispersion around the expected value 0. The model equation (2.58) becomes

$$\frac{\partial X(t)}{\partial t} = F(\, X(t)\,) + W(t) \tag{2.75}$$

The interest of such an assumption is that i) because $Q = 0$ the direct and adjoint equations are not coupled, except at the initial time t_0, ii) at any time t the state x_t is uniquely defined by the initial state x_{t_0}. Therefore, if N is the dimension of the vector x_{t_0}, these N degrees of freedom are enough to fully described the complete model trajectory x_t, $t \geq t_0$ and the minimization problem has now the dimension N instead of $N \times K$, where K is the number of model time steps between $[t_0, t_R]$, as in the Kalman smoother. In addition to this reduction of control variables, the direct and adjoint equations can be integrated separately, but sequentially since the direct model trajectory defines the adjoint operator. This is very suitable for iterative algorithms.

The optimality system corresponding to the 4D-VAR approximation is:

$$x'_{t_0} = x_b + B\lambda_{t_0} \tag{2.76}$$

$$\frac{\partial x'}{\partial t} = F(x'_t) \tag{2.77}$$

$$\lambda_{t_R} = 0 \tag{2.78}$$

$$-\frac{\partial \lambda}{\partial t} = \frac{\partial F}{\partial x}^T \lambda_t - \frac{\partial H}{\partial x}^T R_t^{-1}(y_t - H(x'_t)) \tag{2.79}$$

which can be solved iteratively using a classical descent algorithm.

2.3.2 A discretized form of adjoint model

Until now we used notations which are more consistent with that in the estimation theory. Now we will use notations which are more accustomed to the meteorological community. We change the notation so that a realization of a vector and an operation will use bold face lower and upper case letters respectively and the adjoint variable of a variable $\mathbf{x}$ will be represented by $\hat{\mathbf{x}}$ instead of λ as used before.

The discretized form of the numerical model equation (2.75) can be written as

$$\mathbf{x}(t_r) = \mathbf{Q}_r(\mathbf{x})\mathbf{x}_0, \tag{2.80}$$

and the cost function in (2.60) can be written as:

$$J(\mathbf{x}_0) = 1/2(\mathbf{x}_0 - \mathbf{x}_b)^T\mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}_b) + 1/2\sum_{r=0}^{n}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r)^T\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r) + J^p. \tag{2.81}$$

where $\mathbf{x}_0$ is the analysis vector on the analysis/forecast grid at time t_0, $\mathbf{x}_r$ is the model forecast at time t_r starting from IC $\mathbf{x}_0$, n is the total number of time levels on which observations are available, $\mathbf{x}_b$ is the forecast background vector, $\mathbf{y}_r$ is a vector of observations at time t_r, $\mathbf{O}_r$ is the observation error covariance matrix of the rth observation time (assuming uncorrelated observation errors in time) and is usually assumed diagonal (i.e., all observations are independent) since the inverse of $\mathbf{O}$, $\mathbf{O}^{-1}$, is required in the definition of J, $\mathbf{B}$ is the background error covariance matrix, $\mathbf{H}_r$ is the transformation of model variables to the observational quantities, and J^p is a penalty term controlling gravity wave oscillations.

Symbolically (2.81) can be written as a sum

$$J = J^b + J^o + J^p = J^b + \sum_{r=0}^{n} (J^o)_r + J^p \tag{2.82}$$

where J^b and J^o are the background and the observation terms, respectively. J^b measures the misfit between the model initial state and all available information prior to the assimilation period, summarized by the background field $\mathbf{x}_b$. J^o measures the distance of the model state from the observations at appropriate times during the assimilation window. The term J^o consists of several individual terms $(J^o)_r$ corresponding to various types of observations at time t_r within the assimilation period.

In order to obtain the optimal IC ($\mathbf{x}_0^*$) that minimizes J in (2.81), the gradient of J with respect to the IC ($\mathbf{x}_0$):

$$\nabla J = \nabla J^b + \nabla J^o + \nabla J^p \tag{2.83}$$

needs to be calculated. The first term ∇J^b in (2.82) can be easily obtained as:

$$\nabla J^b = \mathbf{B}^{-1}(\mathbf{x} - \mathbf{x}_b) \tag{2.84}$$

and the second term ∇J^o in (3.82) requires the adjoint model integration which shall be briefly derived as follows:

Consider the change in the cost function J resulting from a small perturbation $\mathbf{x'}_0$ in IC ($\mathbf{x}_0$):

$$J'^o(\mathbf{x}_0) = J^o(\mathbf{x}_0 + \mathbf{x'}_0) - J^o(\mathbf{x}_0) \tag{2.85}$$

On one hand, J'^0 can be expressed as the directional derivative in the $\mathbf{x}'_0$ direction plus the higher-order terms and is given by

$$J'^o(\mathbf{x}_0) = (\nabla J^o(\mathbf{x}_0))^T \mathbf{x}'_0 + O(\|\mathbf{x}_0\|^2). \tag{2.86}$$

On the other hand, substituting (2.81) into (2.85) we obtain

$$J'^o(\mathbf{x}_0) = \sum_{r=0}^{n} \mathbf{H}_r^T \left(\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r)\right)^T \mathbf{x}'_r + O(\|\mathbf{x}_r\|^2) \tag{2.87}$$

where $\mathbf{x}'_r$ is the forecast difference at time t_r between the perturbed and unperturbed forward nonlinear model integrations resulting from the initial perturbation, $\mathbf{x}'_0$. Equating (2.86) and (2.87), results in

$$(\nabla J^o(\mathbf{x}_0))^T \mathbf{x}'_0 = \sum_{r=0}^{n} \mathbf{H}_r^T \left(\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r)\right)^T \mathbf{x}'_r, \tag{2.88}$$

in which the higher-order terms are neglected.

The forward numerical model (2.80) can be differentiated (perturbed) to obtain a so-called tangent linear model (TLM):

$$\mathbf{x}'(t_r) = \mathbf{P}_r(\mathbf{x})\mathbf{x}'_0, \tag{2.89}$$

which predicts in time the perturbation solution, accurate within the first-order approximation.

Using the symbolic expression of the linear version of the forecast model (2.89), (2.88) becomes

$$(\nabla J^o(\mathbf{x}_0))^T \mathbf{x}'_0 = \sum_{r=0}^{n} \mathbf{H}_r^T \left(\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r)\right)^T \mathbf{P}_r \mathbf{x}'_0. \tag{2.90}$$

In the limit of $\|\mathbf{x}'_0\| \to 0$, (2.90) implies

$$\nabla J^o(\mathbf{x}_0) = \sum_{r=0}^{R} \mathbf{P}_r^T \mathbf{H}_r^T \mathbf{O}_r^{-1} (\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r). \tag{2.91}$$

Therefore, the gradient of the cost function $\nabla J^o(\mathbf{x}_0)$ with respect to the IC $\mathbf{x}_0$ can be obtained as a summation of the following variables:

$$\nabla J^o(\mathbf{x}_0) = \sum_{r=0}^{R} \hat{\mathbf{x}}_0^r, \tag{2.92}$$

where $\hat{\mathbf{x}}_0^r$ is the solution of the following equations:

$$\begin{aligned} \hat{\mathbf{x}}_0^r &= \mathbf{P}_r^T(\mathbf{x})\hat{\mathbf{x}}(t_r) \\ \hat{\mathbf{x}}(t_r) &= \mathbf{H}_r^T\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r), \quad r = 0, 1, \ldots, R, \end{aligned} \tag{2.93}$$

i.e., each $\hat{\mathbf{x}}_0^r$ is obtained by the backward adjoint model integration starting from the "initial" condition $\mathbf{H}_r^T\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r)$ at each time t_r. Eq. (2.93) is the discretized adjoint model of (2.80).

Since both (2.92) and (2.93) are linear, the summation in (2.92) representing $\nabla J^o\,(\mathbf{x}_0)$ may be obtained by a single adjoint model integration extending from time t_R to t_0 with zero "initial" conditions for the adjoint variables at time t_n while the weighted differences

$$(forcing\ term) = \mathbf{H}_r^T\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{y}_r) \tag{2.94}$$

are added to the adjoint variables whenever an observational time $t_r(r = R, R-1, \cdots, 0)$ is reached. Thus a single integration of the adjoint model over the assimilation window can yield the value of the gradient of the cost function with respect to the ICs. The approximate computational cost for one single integration of the adjoint model is normally equivalent to 2 or more of the original nonlinear model integrations for the same length of the integration time, which is of course, much cheaper than the finite-difference approximation to the gradient value.

With the availability of the MM5 ($\mathbf{Q}$), and its adjoint model ($\mathbf{P}^T$), and the adjoint of the observation operator $\mathbf{H}$ ($\mathbf{H}^T$), the values of both J and ∇J can be calculated. One can then employ any unconstrained minimization software to find the minimum (see section 3.8 for details). In 4D-VAR, all observations are used at once to perform the analysis globally. The 4D-VAR can directly assimilate many measurements as long as they can be expressed as a function of the basic model variables. For a specific type of observation, users of MM5 adjoint model may need to develop their own adjoint of the observation operator for assimilating that observation. Problems that may be encountered in adjoint coding are described in Chapter 4.

2.4 Mathematical derivation of various adjoint applications

Having described the usefulness of the adjoint model in 4D-VAR, this section provides a brief review of other different applications of adjoint models and presents the mathematical formulae briefly illustrating how the adjoint model and/or tangent linear model are used in each application. The idea is to provide the theoretical background and to show the wide range of applications which could be offered by the availability of the MM5 adjoint model system.

2.4.1 Parameter estimation

The application of the variational approach to optimally determine model parameters is conceptually similar to that of determining the optimal ICs in 4D-VAR. In the following we will present a brief illustration of the method using the Lagrange multiplier method.

Based on criteria $J(\alpha)$ which either measures distance between the model and observations or describes some balance conditions of some meteorological fields, or both, the parameter estimate is to minimize $J(\alpha)$ by adjusting model parameters α, i.e., find the optimal parameters α^* such that

$$J(\alpha^*) \leq J(\alpha), \quad \forall\alpha. \tag{2.95}$$

In order to explicitly indicate the dependence of the model prediction on model parameters, we rewrite the model equation (2.58) into

$$\frac{\partial \mathbf{x}}{\partial t} = \mathbf{F}(\mathbf{x}, \alpha). \tag{2.96}$$

Due to the dynamical coupling of the state variables to the forcing parameters, the dynamics can be enforced through the use of a Lagrange function constructed by appending the model equations to the cost function as constraints in order to avoid the repeated application of the chain rule when differentiating the cost function. The Lagrange function is defined by

$$L(\mathbf{x}, \alpha, \hat{\mathbf{x}}) = J + < \hat{\mathbf{x}}, \frac{\partial \mathbf{x}}{\partial t} - F(\mathbf{x}, \alpha)) > \tag{2.97}$$

where $\hat{\mathbf{x}}$ is a vector of Lagrange multipliers. The Lagrange multipliers are not specified but computed in determining the best fit.

The gradient of the Lagrange function must be zero at the minimum point. This results in the following first order conditions:

$$\frac{\partial L}{\partial \mathbf{x}} = 0. \tag{2.98}$$

$$\frac{\partial L}{\partial \hat{\mathbf{x}}} = 0 \tag{2.99}$$

$$\frac{\partial L}{\partial \alpha} = 0 \tag{2.100}$$

The solution of (2.98)-(2.100) is called stationary point of L.

Substituting L into the above equations we obtain:

$$\frac{\partial \mathbf{x}}{\partial t} = F(\mathbf{x}, \alpha) \quad \text{(nonlinear model)} \tag{2.101}$$

$$-\frac{\partial \hat{\mathbf{x}}}{\partial t} = -\left(\frac{\partial F(\mathbf{x})}{\partial \mathbf{x}}\right)^T \hat{\mathbf{x}} + \frac{\partial J}{\partial \mathbf{x}} \text{(adjoint model)} \tag{2.102a}$$

$$\hat{\mathbf{x}}|_{t=t_R} = 0. \tag{2.102b}$$

and

$$\frac{\partial J}{\partial \alpha} + \int_{t_0}^{t_R} < \hat{\mathbf{x}}, \frac{\partial F(\mathbf{x}, \alpha)}{\partial \alpha} > dt = 0. \tag{2.103}$$

An important relation between the gradient of the cost function with respect to parameters α, $\partial J/\partial \alpha$, and the partial derivative of the Lagrange function with respect to the parameters is

$$\nabla_\alpha J(\alpha) = \frac{\partial L}{\partial \alpha}|_{\text{at stationary point}}, \tag{2.104}$$

i.e.,

$$\nabla_\alpha J(\alpha) = \frac{\partial J}{\partial \alpha} + \int_{t_0}^{t_R} < \hat{\mathbf{x}}, \frac{\partial F(\mathbf{x}, \alpha)}{\partial \alpha} > dt \tag{2.105}$$

Comparing (2.102) with (2.67)-(2.68) we observe that both variational data assimilation and parameter estimate employs the same adjoint equation model, which is used to efficiently compute both the gradient of the cost function with respect to the model IC or the gradient of the cost function with respect to model parameters.

2.4.2 Adjoint sensitivity analysis

In sensitivity analysis studies, the model output of interest is usually referred to as the system's *response*, instead of calling it as a cost function as in data assimilation and parameter estimate. Sensitivity is a measure of the effect of changes in a given input parameter on a selected response. In this subsection, we will consider sensitivity of a response to both model ICs and model parameters for completeness.

Consider, for example, a functional response $R(\mathbf{x}, \alpha)$ of the form

$$R(\mathbf{x}, \alpha) = \int_{t_0}^{t_R} r(t; \mathbf{x}, \alpha) dt \tag{2.106}$$

where $r(t; \mathbf{x}, \alpha)$ depends on model variables $\mathbf{x}$, the parameters α, and the time interval $[t_0, t_R]$ represents the selected time window, where $(t_R - t_0)$ is the time interval of most interest.

The most general definition of the sensitivity of a response to variations in the system parameters is the Gateau-(G-)differential:

$$VR(\mathbf{x}_0, \alpha; \mathbf{x'}_0, \alpha') = \int_{t_0}^{t_R} r'_{\mathbf{x}} \cdot \mathbf{x}' dt + \int_{t_0}^{t_R} r'_{\alpha} \cdot \alpha' dt, \tag{2.107}$$

where

$$r'_{\mathbf{x}} \equiv \left\{ \left(\frac{\partial r}{\partial x_1}, \ldots, \frac{\partial r}{\partial x_P} \right) \right\}_{(\mathbf{x}_0, \alpha)}, \tag{2.108a}$$

$$r'_{\alpha} \equiv \left\{ \left(\frac{\partial r}{\partial \alpha_1}, \ldots, \frac{\partial r}{\partial \alpha_N} \right) \right\}_{(\mathbf{x}_0, \alpha)}; \tag{2.108b}$$

the subscript N is the dimension of the model parameters and P is the dimension of the model variable $\mathbf{x}$.

When $R(\mathbf{x}, \alpha)$ is *continuous* in $\mathbf{x}$ and α, the total variation of R is given by

$$R(\mathbf{x}_0 + \mathbf{x}', \alpha + \alpha') - R(\mathbf{x}_0, \alpha) = VR(\mathbf{x}_0, \alpha; \mathbf{x'}_0, \alpha') + O[||\mathbf{x'}_0||^2] + O[||\alpha'||^2] \tag{2.109}$$

i.e., $VR(\mathbf{x}_0, \alpha; \mathbf{x'}_0, \alpha')$ is linear in $\mathbf{x'}_0$ and α'. If R or its derivatives are discontinuous the G-differential still has meaning.

The simplest and perhaps the most common procedure for sensitivity analysis of a model consists of varying selected input parameters, rerunning the code, and recording the corresponding changes in the response calculated by (2.106). The model parameters responsible for the largest relative changes in the response are classified to be the most important. For complex models, though, the large amount of computing time needed by such recalculations severely restricts the scope of this procedure.

Examining (2.107) we observe that in order to obtain the values of sensitivity of the response, we should know the value of $\mathbf{x}'$ in the time window $[t_0, t_R]$, which is the perturbed nonlinear model solution, i.e., the following TLM solution.

$$\frac{\partial \mathbf{x}'}{\partial t} - \frac{\partial F}{\partial \mathbf{x}}\mathbf{x}' = \frac{\partial F}{\partial \alpha}\alpha' \tag{2.110a}$$

$$\mathbf{x}'|_{t=t_0} = \mathbf{x}'_0 \tag{2.110b}$$

The TLM in (2.110) is different from the TLM in (2.89) in which the model parameters are fixed.

However, when the dimension of the initial state vector and the number of parameters is large, the computational cost of calculating the first term in (2.107) is very high (we have to run the TLM (2.110) P times to obtain all the components of $\mathbf{x}'$, where P is the dimension of the model state variables). Therefore, we eliminate $\mathbf{x}'(t)$ by using the adjoint formulation.

Equation (2.110a) can be rewritten as

$$\mathbf{L}\mathbf{x}' = \frac{\partial F}{\partial \alpha}\alpha' \tag{2.111}$$

where $\mathbf{L} = \partial/\partial t - \partial F/\partial \mathbf{x}$.

The adjoint operator $\mathbf{L}^*$ of the operator $\mathbf{L}$ is defined through the relationship

$$\int_{t_0}^{t_R} \mathbf{x}' \cdot (\mathbf{L}^*\hat{\mathbf{x}})dt \equiv \int_{t_0}^{t_R} \hat{\mathbf{x}} \cdot (\mathbf{L}\mathbf{x}')dt - [\mathbf{x}' \cdot \hat{\mathbf{x}}]_{t_0}^{t_R}, \tag{2.112}$$

where $\hat{\mathbf{x}}$ is at this stage an arbitrary column vector of dimension P.

Using the adjoint model solution satisfying the following equations:

$$\mathbf{L}^*\hat{\mathbf{x}} = r'_{\mathbf{x}}, \tag{2.113a}$$

$$\hat{\mathbf{x}}(t_R) = 0, \tag{2.113b}$$

we obtain

$$\int_{t_0}^{t_R} r'_{\mathbf{x}} \cdot \mathbf{x}'dt = \int_{t_0}^{t_R} \mathbf{L}^*\hat{\mathbf{x}} \cdot \mathbf{x}'dt = \int_{t_0}^{t_R} \hat{\mathbf{x}} \cdot (\mathbf{L}\mathbf{x}')dt + \mathbf{x}'_0 \cdot \hat{\mathbf{x}}_0. \tag{2.114}$$

Substituting (2.111) into (2.114) we obtain

$$\int_{t_0}^{t_R} r'_{\mathbf{x}} \cdot \mathbf{x}'dt = \int_{t_0}^{t_R} \hat{\mathbf{x}} \cdot (\frac{\partial F}{\partial \alpha}\alpha')dt + \mathbf{x}'_0 \cdot \hat{\mathbf{x}}_0. \tag{2.115}$$

With the use of (2.115), (2.107) can be written as

$$VR = \int_{t_0}^{t_R} r'_\alpha \cdot \alpha' dt + \int_{t_0}^{t_R} \hat{\mathbf{x}} \cdot (\frac{\partial F}{\partial \alpha} \alpha')dt + \mathbf{x}'_0 \hat{\mathbf{x}}_0, \tag{2.116}$$

which is the adjoint formulation for sensitivity analysis of a response functional (2.106).

Comparing (2.107) and (2.116) we see that the main advantage of the adjoint formulation is that (2.116) is independent of $\mathbf{x}'(t)$. Thus, (2.116) replaces the time integration of TLM with the calculation of a quadrature $(\partial F/\partial \alpha)$, an operation much cheaper to perform when the number of the model parameters is large. The adjoint variable $\hat{\mathbf{x}}(t)$ is the solution of the adjoint equations (2.113), which are *independent* of $\mathbf{x}'(t)$ and α'. Therefore, a *single* adjoint model calculation suffices to obtain the sensitivities of one functional response to all the model parameters' variations. Since the forcing term, r'_x, in the adjoint model (2.113) depends on the functional defining the response, for each response the adjoint equations model must be integrated anew.

It is obvious that for models that involve a large number of parameters and comparatively few responses, sensitivity analysis can be performed very efficiently by using deterministic methods based on adjoint functions.

If one doesn't consider model parameters as control variables, the first two terms in (2.116) disappear and the sensitivity of a functional response (2.106) is simply the product of the initial perturbation vector and the adjoint variable vector at t_0 resulting from a backward integration of the adjoint model (2.113).

Comparing (2.68), (2.103) and (2.113), we find that in the sensitivity study, we use the same adjoint model as was used in variational data assimilation and parameter estimate experiments. It is the adjoint model, as a computational tool, that makes solving the large-dimension variational data assimilation, parameter estimate and sensitivity study possible and effective.

2.4.3 Singular vectors

Singular vectors (SV's) denote perturbations obtained within linear theory by maximizing the growth of a chosen norm over a finite time interval $(t - t_0)$. Here, t_0 is the initial time. Such perturbations are also called "optimal perturbations."

The calculation of SV's requires *a prior* definition of an inner product in the linear space of the perturbations. In other words, a norm has to be chosen, in accordance with the specific dynamical problem at hand. Different norms can yield dramatically different structures of SV's. Norms commonly used in the literature are the "total energy norm", the "kinetic energy norm", the "enstrophy norm" and the Euclidean "L_2 norm."

Three major areas of application of SV's can be found in the literature:

1. *Study of the predictability of atmospheric flows*:

   Lorenz (1965) suggested that the growth of forecast errors could be conveniently expressed in terms of the growth of the SV's of the forecast error norm. As a follow-up, for the last few years, the ECMWF has been routinely using SV's to construct the center's sets of initial perturbations for Ensemble Forecasting (Molteni et al., 1996).

2. *Study of the instability properties of atmospheric and oceanic flows*:

   SV's provide an alternative to the more "classic" concept of normal modes (Farrell, 1982 and Farrell, 1989).

3. *Adaptive observations*:

   The use of SV's has been proposed to identify (dynamically-sensitive) regions where observations are much needed in order to more accurately determine the IC for model forecasting (Palmer et al., 1998).

We present the mathematical formulas pertinent to the calculation of SV's (Buizza et al., 1993). Let us denote the inner product between two arbitrary vectors $\mathbf{x'}_1$ and $\mathbf{x'}_2$ by $< \mathbf{x'}_1 , \mathbf{x'}_2 >_E$, where the subscript E stands for the chosen norm. Then, in the "E-sense" the norm of the state vector $\mathbf{x}'(t)$ is given by:

$$\| \mathbf{x}'(t) \|_E^2 = < \mathbf{x}'(t) , \mathbf{x}'(t) >_E \quad . \tag{2.117}$$

This is the quantity which we intend to maximize at time t! The time interval $(t - t_0)$ is also referred to as *the optimization time.*

Substituting the TLM expression (2.89) into (2.117), we can write:

$$\| \mathbf{x}'(t) \|_E^2 = < \mathbf{P}\mathbf{x'}_0 , \mathbf{P}\mathbf{x'}_0 >_E \ , \tag{2.118}$$

or

$$\| \mathbf{x}'(t) \|_E^2 = < \mathbf{P}^{*E}\mathbf{P}\mathbf{x}'_0 \,, \mathbf{x}'_0 >_E \,, \tag{2.119}$$

where $\mathbf{P}^{*E}$ is the adjoint of $\mathbf{P}$ with respect to the E-norm. We note that all our previous derivations of adjoint model were under the Euclidean L_2 norm, which is the norm we always use unless specified otherwise.

It is not difficult to show (Buizza et al., 1993) that the perturbation which maximizes the norm (2.117) is the eigenvector of the self-adjoint operator $(\mathbf{P}^{*E}\mathbf{P})$ of largest eigenvalue, i.e., we must solve the eigenvalue problem:

$$(\mathbf{P}^{*E}\mathbf{P})\, \nu_i(t_0) = \sigma_i^2\, \nu_i(t_0) \,. \tag{2.120}$$

Also, the eigenvector associated with the second largest eigenvalue can be shown to produce the second largest amplification of norm, and so on. Due to the self-adjointness of $(\mathbf{P}^{*E}\mathbf{P})$, the SV's $\{\nu_i\}$ form a complete and orthogonal set at the initial time t_0, and their eigenvalues $\{\sigma_i^2\}$ are real. The latter are also positive, which follows from the fact that the norm is introduced through a positive definite matrix. Less obvious is the fact that the eigenvectors form an orthogonal basis at the final time as well (e.g., Noble and Daniel 1977). The eigenvectors $\{\nu_i\}$ are known in linear algebra as the (right-hand) singular vectors (SV's) of the matrix $\mathbf{P}$, with correspondent singular values $\{\sigma_i\}$.

In practice, it is useful to relate the inner product defined in the (general) E-norm with that defined in the Euclidean L_2 norm. This is because we intend to use the adjoint of the forward linear model in order to compute the SV's. With few exceptions, adjoint models are coded in the Euclidean L_2 norm.

For two arbitrary vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, one can write:

$$< \mathbf{x}_1 \,, \mathbf{x}_2 >_E = < \mathbf{x}_1 \,, E\, \mathbf{x}_2 >_{L_2} \,, \tag{2.121}$$

where E is a matrix of weights derivable from the analytic expression for the quantity used to introduce the norm. This point will be made more clear shortly when we treat an example. It is easy to show (Buizza et al., 1993) that the adjoint matrix $\mathbf{P}^{*E}$ in the E-norm is related to the adjoint matrix $\mathbf{P}^*$ in the L_2 norm through the expression:

$$\mathbf{P}^{*E} = E^{-1}\mathbf{P}^*E \,. \tag{2.122}$$

Equation (2.120) is equivalent to seeking the eigenvectors and eigenvalues of:

$$K = E^{-1}\mathbf{P}^{*}E\mathbf{P} \ . \qquad (2.123)$$

Example: "Approximate Energy Norm"

We use the following quadratic expression as an approximation for the total perturbation energy $\mathcal{E}$ of the nonhydrostatic dry version of the MM5 (Bannon, 1995):

$$\mathcal{E} = \int_{\sigma}\int_{x}\int_{y} \left\{ \left(\frac{u'^2 + v'^2 + w'^2}{2} \right) + \frac{1}{2}\frac{g^2}{\overline{N}^2}\frac{\theta'^2}{\overline{\theta^2}} + \frac{1}{2\overline{\rho}^2}\frac{p'^2}{c_s^2} \right\} \left(\frac{\partial p}{\partial \sigma} \right) dx dy d\sigma \ , \qquad (2.124)$$

where the first term on the right-hand side represents the kinetic energy, the second term the available potential energy and the last term the elastic energy. The primes denote perturbation quantities, θ' is the perturbation potential temperature and $\overline{N}^2$ and c_s are reference values for the Brunt-Väisälä frequency and the speed of sound in the basic state, respectively. Also, $\overline{\rho}$ is a reference value for the density. For convenience, we replace the temperature with the potential temperature in the state vector.

If we represent the state vector as $\mathbf{x} = \left(\{u'\}, \{v'\}, \{w'\}, \{\theta'\}, \{p'\}\right)^T$, where the curly brackets represent row sub-matrices made up of all the grid point values of the corresponding model variable, then E is given by a diagonal matrix, whereby its elements are easily derivable from (2.124), simply by imposing that:

$$\mathcal{E} = < \mathbf{x}, E\mathbf{x} >_{L_2} \ . \qquad (2.125)$$

The Lanczos Algorithm

For large systems, as is the case for most of the primitive equation models, finding the solution to (2.120) by means of standard eigenanalysis routines, such as those from EISPACK, is prohibitive. It is then common to resort to the Lanczos algorithm (e.g., Golub and Loan 1989) to compute the leading SV's and singular values. The algorithm does not access directly the elements of the matrix, which is computationally prohibitive in terms of memory and computing time. It involves partial tridiagonalizations of the matrix, whereby information about the extremal eigenvalues emerges long before the tridiagonalization is complete. No intermediate full sub-matrices are generated in the process. In our applications, the Lanczos routines are coupled to the forward and adjoint linear models, and a series of iterations is performed to yield the leading SV's and singular values with the desired accuracy.

The Lanczos routine at our disposal works only for symmetric matrices. It is therefore necessary to transform K, as given by (2.123) into, a symmetric matrix, by means of a coordinate transformation (Buizza et al., 1993). This is achieved by writing:

$$\nu = E^{-1/2}\,\tilde{\nu}\,. \tag{2.125}$$

A new eigenvalue problem is obtained through this transformation:

$$\tilde{K}\,\tilde{\nu}_i = \sigma_i^2\,\tilde{\nu}_i\,, \tag{2.126}$$

where

$$\tilde{K} = E^{1/2}\,K\,E^{-1/2} = E^{-1/2}\,\mathbf{P}^*\,E\,\mathbf{P}\,E^{-1/2} \tag{2.127}$$

is a symmetric matrix.

In practice, the user is required to supply a subroutine which, given an arbitrary vector, returns the product of $\tilde{K}$ with that vector. We see from (2.127) that this subroutine must perform the following sequence of operations:

1. multiply the state vector with matrix $E^{-1/2}$,

2. integrate the resulting vector on the forward linear model from t_0 to t. This accounts for the application of the forward propagator between t_0 and t,

3. multiply the resulting vector with matrix E,

4. integrate the resulting vector on the adjoint model backward from t to t_0. This accounts for the application of the adjoint of the forward propagator in the Euclidean L_2 norm, and

5. multiply the resulting vector with matrix $E^{-1/2}$.

2.4.4 Normal modes and adjoint modes

The TLM (2.89) can be used to calculate the normal modes that grow on a given basic state, as described below:

2.4.4a. Frozen Basic States

For a resting basic state, i.e., for $\mathbf{x}(t) \equiv \mathbf{x}_0 \; \forall t$, model solutions to the TLM

$$\frac{\partial \mathbf{x}'(t)}{\partial t} = \mathbf{L}(\mathbf{x}(t))\mathbf{x}'(t), \qquad \mathbf{L}(\mathbf{x}(t)) = \frac{\partial F}{\partial \mathbf{x}}(\mathbf{x}(t)) \tag{2.128}$$

that possess an exponential time dependence of the amplitude can be obtained through the ansatz $\mathbf{x}'(t) = e^{\sigma t}\mathbf{z}$, since the linear operator $\mathbf{L}$ becomes independent of time. This leads to the eigenvalue problem:

$$\mathbf{L}\mathbf{z} = \sigma \mathbf{z}, \tag{2.129}$$

where $\mathbf{z} = \mathbf{z}_r + i\mathbf{z}_i$ is the eigenvector and $\sigma = \sigma_r + i\sigma_i$ is the eigenvalue. The real part of σ, σ_r, is the growth rate and the imaginary part, σ_i, is the frequency. The n^{th} normal mode is defined as:

$$\mathbf{Z}_n = Re\{\mathbf{z}_n e^{\sigma_n t}\} = e^{\sigma_{nr} t}\{\mathbf{z}_{nr}\, cos(\sigma_{ni} t) - \mathbf{z}_{ni}\, sin(\sigma_{ni} t)\}\,. \tag{2.130}$$

Equation (2.129) can be solved by two distinct approaches:

1. *Relatively small systems:* Solving (2.129) is a straightforward task when the system is of a "reasonably small" dimension, i.e., when the matrix $\mathbf{L}$ can be stored in the computer memory and standard eigenproblem solvers (e.g. routines from NAG) can be used. We note that $\mathbf{L}$ can be found very easily with the help of the (time-stepping) TLM. By setting the l^{th} element of the vector of the IC $\mathbf{x'}_0$ to 1, zeroing all the other elements, and making a one-time-step integration, we obtain the tendency $d\mathbf{x}/dt$, which is the l^{th} column of $\mathbf{L}$. Repeating this process N times, where N is the dimension of the system, we obtain the matrix $\mathbf{L}$. The procedure thus consists of generating a column of $\mathbf{L}$ at a time.

   For frozen basic states, the normal modes $\{\mathbf{Z}_n\}$ represent shape-preserving solutions. When integrated on the TLM, the shapes of these perturbations repeat themselves after every period $2\pi/\sigma_{ni}$.

2. *Large systems:* The use of the method outlined above becomes prohibitive when we are interested in systems of large dimensions, as is the case in most applications of the MM5. In such cases, however, we can still compute the leading normal modes and corresponding eigenvalues by resorting to an algorithm which does not access

directly the elements of the matrix. The Lanczos method, which is very successful in computing singular vectors (see section on singular vectors), does not appear to be very useful in dealing with asymmetric matrices. We recall that, in general, **L** is an asymmetric matrix. Anderson (1991) used with success the method proposed by Goldhirsch et al. (1987) in his calculations of the normal modes of a barotropic model. The method can be applied to large asymmetric matrices and consists of reducing the large $N \times N$ original problem to a smaller $K \times K$ problem ($K << N$). This is accomplished iteratively with the help of the TLM, a standard eigenproblem solver and an orthogonalization routine. The method consists of generating $K << N$ linearly independent vectors, which, when integrated a number of times on the TLM, and orthogonalized, will converge to a certain sub-matrix of dimension $K \times K$. The eigenvectors of that submatrix are the K most unstable eigenvectors of L. For detailed derivation of this method, please see Goldhirsch et al. (1987).

2.4.4b. Time Varying Basic States

The solution to (2.128), subject to the IC $\mathbf{x}'(t_0) = \mathbf{x}'_0$, can be written as (2.89), where $\mathbf{P}(t)$ is called the forward propagator between times t_o and t.

Following Frederiksen (1997), the finite-time normal mode eigenvectors of a time dependent basic state, between times t_o and $t_o + \tau$, can be defined as the eigenvectors of the forward propagator for this time interval. The finite-time normal modes are defined in a manner analogous to (2.130). If the n^{th} eigenvalue of the forward propagator is $\lambda_{nr} + i\,\lambda_{ni}$, then the growth rate and phase frequency of the n^{th} finite-time normal mode are conveniently defined as $\sigma_{nr} = \frac{1}{2\tau} ln(\lambda_{nr}^2 + \lambda_{ni}^2)$ and $\sigma_{ni} = \frac{1}{\tau} arctan \frac{\lambda_{ni}}{\lambda_{nr}}$, respectively.

The leading finite-time normal modes and corresponding growth rates and phase frequencies can be computed using the method described in 2.4.4a.

2.4.4c. Adjoint Modes

The adjoint eigenvectors are the eigenvectors of the adjoint of the matrix of the linear problem (for time invariant basic states) or of the adjoint of the forward propagator (for time evolving basic states). In the latter case, the terminology **finite-time adjoint eigenvectors** is more appropriate. The adjoint modes (and finite-time adjoint modes) are defined in a manner analogous to (2.130).

The computation of adjoint eigenvectors involves the definition of a norm. This is done by introducing a matrix of weights, **E** (see section 2.4.3),

For time invariant basic states and when the system is not too large, one can find **L** as described in point 1 of 2.4.4a., and then calculate its adjoint in the chosen E-norm. The adjoint eigenvectors and respective eigenvalues (which are complex conjugates to those of **L**) can be found by using a standard eigenanalysis routine.

For large systems, or for time evolving basic states (irrespective of the dimension of the problem), one can resort to the method described in point 2 of 2.4.4a, i.e., the adjoint of the TLM can be coupled to the Goldhirsch et al. routine.

We note that the adjoint of the MM5 TLM, which is coded in the Euclidean L_2 norm, can be used to obtain the adjoint modes corresponding to any arbitrary norm by assigning a proper weighting coefficient to the inner product which defines the norm (see section 2.4.3 on Singular Vectors for the relation between the adjoint of an operator with respect to any E-norm and the adjoint with respect to the Euclidean L_2-norm).

Within the context of atmospheric dynamics, adjoint modes are useful in their connection to the normal modes. For perturbations normalized to have the same initial amplitude, adjoint modes represent the best initial perturbations to excite the normal modes in the limit of time approaching infinity (Farrel, 1982 and 1989). The concept of "time approaching infinity" is problem dependent (De Pondeca et al., 1998). In blocking, for instance, it can mean 2 to 3 days or even less. If one believes that a certain aspect of the atmospheric (or oceanic) dynamics simply represents the excitation of a certain normal mode, then one way of searching for the geographical regions where perturbations most contribute to the excitation of that normal mode is to look at its adjoint mode. For instance, people found in several studies of the dynamics of mid-latitudes that the adjoint mode of the fastest growing normal mode has its amplitude concentrated in the equatorial region. This makes them suspect that those particular mid-latitude phenomena were excited by perturbations that appeared in the tropics!

2.4.5 Inverse tangent linear model

Predictability studies suggest that improvements in the estimation of the initial state offer the most promising path to more accurate forecasts, although there is still room for benefits from model improvement (Simmons , 1995). Recently, there has been considerable interest in the investigation of the sensitivity of forecast errors to ICs. Both the adjoint technique (Rabier et al., 1996) and the inverse-linear method (Pu et al., 1996) were used to find the initial perturbation which reduces the 1-2 day forecast errors, defined as the difference between the forecast and analysis verified at the same time. In this section, we will briefly describe the inverse-linear method, how it works, and its limitations.

Given a finite initial perturbation $\mathbf{x'}_0$ to the IC $\mathbf{x}_0$, the evolution of the forecast difference between the perturbed and unperturbed ICs can be approximated as

$$\mathbf{Q}_t(\mathbf{x}_0 + \mathbf{x'}_0) - Q_t(\mathbf{x}_0) = \mathbf{P}_t\mathbf{x'}_0 + O(\|\mathbf{x'}_0\|^2) \qquad (2.131)$$

As indicated in Zou et al. (1997), the TLM (the first term in the right-hand-side of above equation) approximates the nonlinear difference very well for short-range prediction (up to one day), i.e., the second-order term in (2.131) can be ignored. If we approximate the inverse of the TLM, $\mathbf{P}_t^{-1}$, by integrating the TLM backward in time, we should be able to approximately recover the initial perturbation $\mathbf{x'}_0$ from two nonlinear model solutions at time t:

$$\mathbf{x'}(t) = \mathbf{P}_t^{-1}\left(\mathbf{Q}_t(\mathbf{x}_0 + \mathbf{x'}_0) - Q_t(\mathbf{x}_0)\right) = \mathbf{P}_t\mathbf{x'}_0 + O(\|\mathbf{x'}_0\|^2) \qquad (2.132)$$

If we substitute the perturbed nonlinear model forecast $\mathbf{Q}_t(\mathbf{x}_0 + \mathbf{x'}_0)$ by the verifying analysis at time t, $\mathbf{x}_t^a$, we can obtain the "initial error estimate". It is the solution obtained when the short-range forecast error was traced back to the initial time. The advantage of using the inverse-TLM is that it is cheaper, and its result does not depend on the choice of the norm used in the definition of the error cost function as it does using the adjoint technique.

The limitation of the inverse-TLM method is from the fact that complex numerical models are, in general, not reversible due to the existence of diabatic and dissipative processes such as heating, friction, diffusion, precipitation, and cloud formation, non structure-preserving spatial and temporal discretization schemes used in the numerical models. However, experiences by both Pu et al. (1996) and Wang et al. (1996) showed that an adiabatic model with simple surface friction and vertical diffusion worked well. The inverse-TLM is realized by running such a simplified TLM with a negative time step, and reversing the sign of friction and diffusion terms.

Although the "inverse" TLM is cheaper and its result does not depend on the choice of the norm, its application to sensitivity analysis and data assimilation is still very limited in the following sense: (i) "observations" have to be available for all model variables and on all grid points; (ii) indirect observations can not be easily included due to the need of the inverse of the observation operators which do not necessarily exist; (iii) physical processes cannot be easily included in the "inverse"TLM.

2.4.6 Incremental 4D-VAR approach

A 4D-VAR experiment requires many more computations (up to 100 times) than 3DVAR does. To achieve this number of computations within operational time constraint would require a significantly faster computer or a substantial algorithm improvement or both (Courtier *et al.* 1994). An approximation of the 4D-VAR, namely the incremental approach, similar to the linearization underlying the extended Kalman filter equations, seems to achieve the amount of the computational reduction in a way consistent with the non-linear estimation theory.

The incremental approach was first introduced by Courtier *et al.* (1994). The incremental approach considers a perturbation or an "increment" instead of the full model state. For a small perturbation on the ICs, we can linearize the problem along a reference trajectory and use the tangent model (with a lower resolution, simpler or no physics) instead of the model itself.

In the incremental approach, the variable is not the ICs, but rather the perturbation with which it deviates from the background term:

$$\delta \mathbf{x}(t_0) = \mathbf{x}(t_0) - \mathbf{x}_b \tag{2.134}$$

Assuming that the perturbation is small, the state vector $\mathbf{x}(t)$ can be approximated by the the tangent model along the reference model trajectory $\mathbf{Q}_t(\mathbf{x})\mathbf{x}_b$. In other words, during the minimization process, the model state will be given by the Taylor expansion near the background state:

$$\begin{aligned} \mathbf{x}(t) &= \mathbf{Q}_t(\mathbf{x})\mathbf{x}(t_0) = \mathbf{Q}_t(\mathbf{x})[\mathbf{x}_b + \delta\mathbf{x}(t_0)] \\ &\simeq \mathbf{Q}_t(\mathbf{x})\mathbf{x}_b + \mathbf{P}_{t_0,t} * \delta\mathbf{x}(t_0) \end{aligned} \tag{2.135}$$

In turn, the model observation counterpart is then given by:

$$\begin{aligned} H_i[\mathbf{x}(t)] &= H_i[\mathbf{Q}_t(\mathbf{x})\mathbf{x}(t_0)] \simeq H_i[\,\mathbf{Q}_t(\mathbf{x})\mathbf{x}_b + \mathbf{P}_t * \delta\mathbf{x}(t_0)\,] \\ &\simeq H_i[\mathbf{Q}_t(\mathbf{x})\mathbf{x}_b] + H_i' * \mathbf{P}_t * \delta\mathbf{x}(t_0) \end{aligned} \tag{2.136}$$

and the cost function (2.81) is modified into:

$$\begin{aligned} J &= \frac{1}{2}\,\delta\mathbf{x}(t_0)^T B^{-1}\delta\mathbf{x}(t_0) + \\ & \frac{1}{2}\sum_{i=1}^{N}(Y_i - H_i[\mathbf{Q}_{t_i}(\mathbf{x})\mathbf{x}_b] - H_i' * \mathbf{P}_{t_i} * \delta\mathbf{x}(t_0))^T O^{-1} \\ & \qquad (Y_i - H_i[\mathbf{Q}_{t_i}(\mathbf{x})[\mathbf{x}_b]] - H_i' * \mathbf{P}_{t_i} * \delta\mathbf{x}(t_0)) \end{aligned} \tag{2.137}$$

The interest of such a formulation is that, once the reference trajectory $\mathbf{Q}_t\mathbf{x}_b$ and the predicted observation $H_i\mathbf{Q}_t\mathbf{x}_b$ are computed, they are kept constant during the minimization. Also, in such a formulation the problem is now linear and the cost function is quadratic. Approximating the full problem by a quadratic problem has theoretical advantages since the solution in principle is guaranteed to be unique. From a statistical point of view, this linearized solution corresponds to the classical extended Kalman filter when the model error is negligible.

If the tangent model is a good approximation of the real model, the linear solution is close to the real solution. This implies that it will be the best if the tangent linear model is used within the time range of its validity. Several studies (Lacarra & Talagrand 1988, Errico *et al.* 1993) have shown that, for the model dynamics, the TLM is a quite robust approximation to its nonlinear solution when it is used in its time limit of validity ($\leq$ 24 hours in mesoscale studies and 2-3 days for large scales). However, when physics is incorporated into the model, nonlinearities increase and linearization is valid with much more restriction. In this case we should modify the previous approach, so as to introduce some full physics model updates in the reference trajectory, or to include some simple physics in the tangent linear model used in the incremental minimization procedure. The new algorithm will then be made of a pair of nested loops. The inner loop is based on the incremental approach described by equations (2.135 and 2.136), the outer loop is simply made of full physics model integrations.

Because the full tangent model integration has a computing cost similar or slightly higher to the direct model integration, the incremental approach, as it is presented, does not really reduce the computing cost of 4D-VAR. However, this approach is susceptible to further cost effective approximations. If, for instance, in the inner loop we use the tangent and the adjoint models at a lower resolution or/and with a degraded physics, computations can be rapidly reduced by an order of magnitude. At the European Center, the use of the adiabatic tangent model at truncation T106 instead of the full physics model at truncation T213 has reduced the computing time by a factor 16 (Courtier *et al.* 1994). Applications of the method to the MM5 adjoint modeling system requires beforehand a review of the acceptable approximations with an estimation of the computing time gain.

The MM5 mesoscale model (Grell et al., 1995) is usually run in a nested domain configuration: a large domain with a low resolution providing the boundary conditions to the smaller high resolution domain under study. The implementation of the 4D-VAR incremental approach to MM5 can take advantage of this specific configuration. It is possible to run the full physics model on the two domains and perform the assimilation with only the tangent model on the larger domain. The main difficulty in this kind of experiment is the interpolation — some conservative properties might be altered when interpolating the low-resolution perturbation onto the high resolution model initial state. This could result in numerical instabilities in the update phase when integrating the full non-linear model.

Different types of physical parameterizations can be selected in the MM5 model. Depending on the selected scheme the computing time can vary a lot. One example is the cumulus convection parameterization schemes, which can have an order of magnitude increase in computational cost. Another example is the parameterization of the planetary boundary layer (PBL) which can also prove to be very time consuming. The different schemes can also have an order of magnitude difference in computing cost. The third example is the explicit microphysical schemes which can differ dramatically. It is interesting to note that, depending on the selected parameterization, some observations, like precipitation, can or cannot be assimilated because of the lack of representation in the model. Therefore, the incremental approach based on a physical approximation may not always be possible.

2.4.7 Optimal control of model error

As mentioned in section 2.2.4, the 4D-VAR algorithm is to find the solution of the data assimilation problem for a case when errors in the forecast model are neglected ($W \simeq 0$ in (2.58)). Thus, in 4D-VAR numerical models are assumed to perfectly represent the atmosphere.

This approximation was made primarily because of the practical necessity. The complete theoretical solution of the data assimilation problem is known in the estimation theory as the *Kalman smoother.* However, such algorithms requires the control of the forecast error at every time-step, and this is computationally far too expensive to be implemented in operational application. The perfect model assumption has the consequence of reducing the optimal control vector of the 4D data assimilation problem to model IC, instead of model IC plus its time evolution. This makes 4D data assimilation computationally feasible. It is an assumption that has certain theoretical justifications. As pointed out by Dee (1995), statistics on the model error are very rare. Studies on the effects of the model error sources on the forecast error in the operational system (Dalcher and Kalnay 1987, Tribbia and Baumhefner 1988) have not produced quantitative information in a form that is useful for data assimilation. In the absence of such information, statistical model input cannot be specified and no meaningful results can be expected, with whatever the amount of computing power, if the required input is either missing or misspecified. It is, therefore, preferable not to specify any error if it cannot be specified correctly. This way, errors in the forecast result exclusively from errors in the initial data, and forecast accuracy is only a function of the predictability limit.

On the other hand, there have been several studies (Boer 1984, Bloom and Schubert 1990) suggesting that model error can sometimes be significant and its effects on forecast error must somehow be accounted for. To overcome the shortage in quantitative information available on the model error, simplifications under the form of assumptions on its shape have been proposed by several authors. These definitions for the model error terms vary from a systematic model bias in a strong constraint formalism (Derber 1989), a systematic model bias in a weak constraint formalism (Wergen 1992), and a slowly time varying stochastic process (Zupanski 1996), to a correlated time varying stochastic process (Daley 1992). Remember that the general case is an uncorrelated time-varying stochastic process. However, only the methods of Derber and Zupanski have been tested in realistic situations and are potentially implementable in the MM5 adjoint modeling system.

To understand how these model error representations differ in terms of information content and computational cost, it is necessary to go back to the original estimation problem of the data assimilation. In the general theory of the Kalman filtering, the model is represented by an equation of the form:

$$\mathbf{x}(t_r) \; = \; \mathbf{Q}_r(x)\mathbf{x}_0 \; + \; \mathbf{w}_r \tag{2.138}$$

where the model error $\mathbf{w}$ figures a pure random white process. The term "white" means that the variations of $\mathbf{w}$ from one time step to the other are completely independent, *i.e* $\mathbf{w}$ is completely unpredictable. This extreme situation corresponds to the minimal level of *a priori* information, the computational cost is therefore maximal, since "all" has to be estimated. In general, the more *a priori* information is specified, the less computation is needed. Another level of information immediately above the white process is the so-called *colored* noise. This level does not provide direct information on $\mathbf{w}$ itself but on its statistical properties. This information is given under the form of a statistical correlation between errors at different time steps, *i.e* $E\{\mathbf{w}_r\mathbf{w}_{r'}^T\} \neq 0$. This is not enough information to allow a prediction of $\mathbf{w}_{r'}$ from $\mathbf{w}_r$, but it can be incorporated in the cost function where it provides some additional constraints in the minimization. Constraint helps in reducing the dimension of the minimization problem. According to Daley (1992), it is certainly at this information level that real meteorological problems exist. A much richer *a priori* information level is found when some predictions of the $\mathbf{w}$ values are possible on limited time intervals: *i.e.* $\mathbf{w}_{r'} = f(\mathbf{w}_r)$ if $|r'-r| < C$ where C is the length of the time interval — this is the slow-time varying representation of the model error used by Zupanski. The next level of information corresponds to the case of a predictable model error over the whole assimilation time period R, *i.e.* $C = R$. This level corresponds to the method proposed by Wergen (1992). If, in addition, we assume the model error to be deterministic we obtain the variational continuous assimilation scheme of Derber. The ultimate level of maximum information occurs when the model error is predictable and known (to be equal to 0), this is the current level of information of the classical 4D-VAR data assimilation algorithm.

Let us examine the Derber and Zupanski model error representations which are, at the present time, the only methods tractable for a complex model such as MM5. The variational continuous assimilation scheme is a control of the model bias. The model error representation consists of the addition of a simple deterministic term ϕ in the model equations. This term can be viewed as a general model bias multiplied by a prescribed time modulating function w_r. Thus, the original "perfect" model equations (2.80) of the 4D-VAR problem are slightly modified into:

$$\mathbf{x}(t_r) \; = \; \mathbf{Q}_r(x)\mathbf{x}_0 \; + \; w_r * \phi \tag{2.139}$$

The time modulation $\mathbf{w}_r$ allows us to weight the contribution of the model errors with their respective time step, since the forecast error is likely to be more influenced by the most recent times steps. This function is specified by the user. In his original paper, Derber (1989) investigated 3 different time weighting functions: i) parabolic: w is a parabola null at the initial time reaching its maximum at the end of the assimilation period; ii) constant: then $w * \phi$ is the global model bias; iii) delta function: w is equal to 1 at the initial time and 0 after, with such a function the estimation reduces to the initial conditions only, this is equivalent to the 4D-VAR data assimilation solution. In addition, the function w_r is normalized so that the sum over all time steps in the assimilation interval is equal to 1.

Since such a model error term is purely deterministic, it has no statistical effects and its contribution in the cost function is null. Thus, the cost function (2.81) defined in the classical 4D-VAR data assimilation problem is still valid. If IC are not controlled, the background information (the first term in (2.81)) should be removed. However, IC and model error can be simultaneously estimated. In that case, the minimization of the cost function is performed with respect to both variables IC and ϕ. The gradient of J with respect to IC is given by:

$$\nabla_{x_0} J = \hat{\mathbf{x}}_0 \tag{2.140}$$

The gradient of J with respect to the model bias error can be derived in a similar way as was done in section 2.4.1; which, in a discrete form, is expressed as:

$$\nabla_{\phi} J = \sum_{r=1}^{R} \mathbf{w}_r \hat{\mathbf{x}}_r \tag{2.141}$$

Compared to the classical variational assimilation procedure, the variational continuous assimilation scheme needs to compute an additional expression (2.141). But, as we can see, the information needed to evaluate this expression consist of the adjoint states $\hat{\mathbf{x}}$ only, which are, in any case, required in the classical scheme. The computational cost of the continuous variational method is therefore equivalent to a classical 4D-VAR data assimilation.

In Zupanski's representation, the model error is a random process which does not vary for a specified time period. Thus, the model was assumed to possess some time intervals during which the error is near constant. Suppose that the model error varies every C time-steps during the assimilation time window consisting of R model time-steps, then the general model equation:

$$\mathbf{x}(t_r) \; = \; \mathbf{Q}_r(x)\mathbf{x}_0 \; + \; \mathbf{w}_r, \;\; r = 1, ..., R \tag{2.142}$$

can be reduced to:

$$\mathbf{x}(t_r) \; = \; \mathbf{Q}_r(x)\mathbf{x}_0 \; + \; \mathbf{w}_m, \;\; r = 1, ..., R, \;\; m = 1, ..., R/C \tag{2.143}$$

for $t_r \in [m\Delta t, \; (m+C)\Delta t]$, where Δt is the model time-step. Thus, the parameters to estimate are the IC $\mathbf{x}_0$ and the R/C vectors $\mathbf{w}_m$.

From a statistical point view, the process $\mathbf{w}$ is correlated in time. This can be expressed as:

$$\begin{aligned} E\{\mathbf{w}_r\mathbf{w}_{r'}^T\} \; = \; E\{\mathbf{w}_m\mathbf{w}_m^T\} \; &= \; \mathbf{\Omega}_m, \;\; if \; (t_r, \; t_{r'}) \in [m\Delta t, \; (m+C)\Delta t] \\ &= \; 0, \qquad if \; |t_r - t_{r'}| > C\Delta t \end{aligned} \tag{2.144}$$

where $\mathbf{\Omega}_m$ is the covariance matrix assumed to be known and diagonal. This statistical *a priori* information should be introduced in the cost function (2.81) which is modified accordingly into:

$$\begin{aligned} J \; = \; 1/2(\mathbf{x}_0 - \mathbf{x}_b)\mathbf{B}^{-1}(\mathbf{x}_0 - \mathbf{x}_b) \; &+ \; 1/2\sum_{r=0}^{r=n}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{d}_r)^T\mathbf{O}_r^{-1}(\mathbf{H}_r(\mathbf{x}_r) - \mathbf{d}_r) \\ &+ \; 1/2\sum_{m=0}^{m=R/C} \mathbf{w}_m^T\mathbf{\Omega}_m^{-1}\mathbf{w}_m \end{aligned} \tag{2.145}$$

Now, the minimization has to be performed, with respect to the IC $\mathbf{x}_0$ and the R/C model error terms $\mathbf{w}_m$. Therefore, in addition to the computation of the gradient $\nabla_{x_0}J$, there are R/C additional gradients$\nabla_{w_m}J$ corresponding to the derivation of the cost function with respect the vectors $\mathbf{w}_m$. The evaluation of these gradients slightly differs from a simple sensitivity calculation, since the additional term in the cost function should also be derived. The method for deriving these expressions is, however, similar and based on the adjoint model. These gradients are given by:

$$\nabla_{w_m}J \; = \; \sum_{c=m}^{m+C}\hat{\mathbf{x}}_c \; + \; \mathbf{\Omega}_m^{-1}\mathbf{w}_m \tag{2.146}$$

Compared to a simple bias, we see that the computation of the model error gradient is very similar, except for the additional computations for the matrix-vector product. For diagonal covariance matrices, this represents a very small amount of additional computations. There are, however, R/C expressions to compute. With the values used in Zupanski's paper of $R = 12h/\Delta t$ and $C = 3h/\Delta t$; this remains of the same order as a classical 4D-VAR assimilation.

This representation of the model error is interesting, since it is an affordable approximation of a time correlated process and it is almost certain (Daley 1992) that, in reality, the model error is serially correlated both in time and in space. However, in the absence of reliable statistics on the model error, the time constant C, which is the key parameter in the method, can only be empirically determined.

2.4.8 Predictability study using adjoint model

It is common knowledge that weather forecasts may fail after a certain period of forecast duration. It has also been observed for a long time by operational centers (Toth and Kalnay 1995) that the loss of skill in the forecast does not occur at the same lead time everyday. One reason for the failure of weather forecasts can be attributed to the imperfection of numerical models in representing the actual atmosphere. However, it has been known since the pioneering work of Lorenz (1963) that this is not the only reason for forecast failure.

As Lorenz showed, the most fundamental cause of forecast failure is that the atmosphere is a chaotic system. A chaotic system can be defined as one whose evolution is sensitive to ICs. This means that an arbitrarily small error in the analysis of the initial state of the atmosphere can have an overwhelming effect in a finite time. The length of atmospheric predictability for forecast failure to occur has been estimated to be between 2 and 4 weeks (Toth 1991). This predictability is inherent to the atmosphere and nothing can be done to push this limit further.

What is possible, however, is to predict how the loss of forecast skill will occur in a forecasting system. By studying how small perturbations, introduced around the initial time, grow; it can be inferred how sensitive the model is and, thus, how robust are the corresponding forecasts. For instance, if two similar initial states lead to very different trajectories, then forecasts will be highly sensitive to initial analysis errors and can be erroneous. In that case, the model predictability will be said to be low. On the contrary, if most of the model trajectories converge to the same state, then this state will constitute a good forecast with a high degree of confidence, since probability for the true atmosphere to be different is very low. This favorable situation corresponds to a high predictability system.

The reason for two trajectories to diverge is that atmospheric states are unstable. Predictability issues arise, thus, when uncertainty in initial data combines with instability. The faster the uncertainty grows, the less predictable the system is. An assessment of the uncertainty present in both the model and the data is therefore a pre-requisite for predictability studies. In order to quantify this uncertainty, perturbations should be chosen so that they can excite all the possible, and at least the unstable modes in the model. The choice of the perturbation is therefore crucial and is still the subject of discussion within the meteorological community. Several techniques have been proposed (see Palmer 1995 for a review), the most popular being the breeding and the singular vectors methods.

The breeding method (Toth and Kalnay 1995) performs a kind of Lyapunov vector computation. A random initial perturbation is generated with a specified amplitude, characteristics of a typical uncertainty in the initial state. Two integrations are run over a specified cycle time (*e.g.* 12h). The first is an integration of the operational weather prediction model from the operational initial state. The second integration is made using the same model, but is initialized by adding the perturbation to the operational analysis. At the end of the integration period, the difference between the two integrations is renormalized using the specified amplitude. The process is repeated for the next cycles, and each time the previous renormalized perturbation is used to generate the new perturbed initial state. In the long run, these perturbations (called breeding vectors) are taken as the most likely error patterns in the forecasting system and they are used to generate the perturbations for the ensemble forecasts.

Singular vectors (see Section 2.4.3), which are computed from the TLM adjoint model, have been recently applied to such study in which singular vectors are used as various elements in Ensemble forecast (Molteni et al., 1996). These vectors represent the most rapidly growing modes in the early stage of the model integration. They are, therefore, the directions along which initial perturbations will likely expand the most during the forecast.

CHAPTER 3: THE MM5 ADJOINT MODEL SYSTEM

This section describes in detail the MM5 adjoint model system. The testing and the performance of the system can be found in the papers of Zou et al. (1995), Kuo et al. (1995), Zou (1996), Zou and Kuo (1996), Kuo et al. (1997), and Zou et al. (1997).

For the past three years, the Mesoscale Prediction Group at the NCAR/MMM division, under the support of the National Science Foundation, the Federal Aviation Administration, and DOE/ARM, has been developing a mesoscale data assimilation system based on the nonhydrostatic version of MM5 and its adjoint. Since MM5 employs the fully compressible system of equations, it is capable of explicitly simulating weather systems from the synoptic scale to the mesoscale and cloud scale. With the availability of more than one option for various physical processes, MM5, its TLM and adjoint model, and the MM5 4D-VAR system based on these models will be a unique tool for mesoscale meteorological research.

As the first step of this effort, the MM5 TLM and adjoint model with complex physics have been developed. In this chapter, we describe the MM5 TLM and adjoint model. Features of the restart of the minimization procedure, the proper handling of disk space for large problems, the infrequent basic-state update, and the minimization procedure are also provided. We also briefly describe how to carry out 4D-VAR and sensitivity experiments using these models and provide a description of the minimum work that the users may need to do in order to run their own case. Principles of the adjoint code development and various examples mostly encountered in the MM5 adjoint model development are provided in Chapter 4.

3.1 MM5 TLM and its adjoint model

A numerical model, such as MM5, used for the study of atmospheric dynamics and physics, is a computer code that uses initial and lateral boundary conditions and the many parameters that define the physical and numerical conditions as independent variables (input). The dependent variables (output) of a numerical model consist of a temporal sequence of meteorological fields produced by the integration. The concept of input (variables) and output (variables) plays an important role in the whole procedure of adjoint model development.

Preceding page blank

MM5 is a limited domain nonhydrostatic finite-difference model (Grell et al., 1994). A time splitting scheme is applied to handle sound waves and fast moving external gravity waves to increase efficiency. It possesses multiple options of physical parameterization. We can write such a model (i.e., MM5) in a general form

$$\frac{\partial \mathbf{x}}{\partial t} = F(\mathbf{x}) \tag{3.1a}$$

$$\mathbf{x}|_{t=t_0} = \mathbf{x}_0 \tag{3.1b}$$

$$\mathbf{x}(t)|_\Gamma = \mathbf{y}(t) \tag{3.1c}$$

where $\mathbf{x}_0$ and $\mathbf{y}(t)$ represent IC (a model state at the initial time t_0), and lateral boundary condition (LBC, a condition that the model state needs to satisfy at the boundary of the domain: Γ), respectively. In MM5, there are six or eight control variables if a microphysical scheme is used. These are 3-dimensional wind u, v, and w, temperature T, relative humidity q_v, pressure perturbation p', cloud water q_c and rain water q_r. The model uses a terrain-following σ-coordinate defined entirely from a reference state $(p_0(z), T_0(z), \rho_0(z))$,

$$\sigma = \frac{p_0 - p_t}{p^*}, \qquad p^* = p_s - p_t, \tag{3.2}$$

where p_s and p_t (p_t=100 mb) are the surface and top reference pressures of the model, respectively. MM5 uses a flux form for advection, and its variables are coupled with p^*, which is a time-independent 2D model constant.

Linearizing the forecast model (3.1) about a nonlinear model trajectory $\mathbf{x}(t)$ and $\mathbf{y}$, we obtain a TLM which can be written as

$$\frac{\partial \mathbf{x}'}{\partial t} = \frac{\partial F(\mathbf{x})}{\partial \mathbf{x}}\mathbf{x}' \tag{3.3a}$$

$$\mathbf{x}'|_{t=t_0} = \mathbf{x}'_0 \tag{3.3b}$$

$$\mathbf{x}'(t)|_\Gamma = \mathbf{y}'(t) \tag{3.3c}$$

where prime represents perturbations of the corresponding variables. Integrating MM5 TLM, starting from a perturbed IC ($\mathbf{x}'_0$) and/or LBC ($\mathbf{y}'_i, i = 1, \ldots, ii$), one obtains a perturbation solution $\mathbf{x}'(t)$ which is accurate to the first-order approximation $O(\|\mathbf{x}'_0\|^2)$ and $O(\|\mathbf{y}'_i\|^2)$; i.e., compared to the true perturbation solution

$$\mathbf{x}'_{\text{true}}(t) = \mathbf{x}(t)|_{(\mathbf{x}_0+\mathbf{x}'_0, \mathbf{y}+\mathbf{y}')} - \mathbf{x}(t)|_{(\mathbf{x}_0, \mathbf{y})}, \tag{3.4}$$

the TLM solution $\mathbf{x}'$ satisfies the following equation:

$$\mathbf{x}'_{\text{true}}(t) = \mathbf{x}'(t) + O(\|\mathbf{x}'_0\|^2) + O(\|\mathbf{y}'_i\|^2). \tag{3.5}$$

The word "adjoint" comes from the inner product in linear space. A linear operator, $\mathbf{L}^*$, is said to be the adjoint of $\mathbf{L}$ if, for all $\mathbf{x}$ and $\mathbf{y}$ in a linear space $\mathcal{S}$,

$$< \mathbf{y}, \mathbf{L}\mathbf{x} > = < \mathbf{L}^*\mathbf{y}, \mathbf{x} > \tag{3.6}$$

where $< \cdot, \cdot >$ represents an inner product. In Euclidean space, $\mathbf{L}^* = \mathbf{L}^T$.

The adjoint model corresponding to (3.3) is (see Section 2.3.1)

$$-\frac{\partial \hat{\mathbf{x}}}{\partial t} = \left(\frac{\partial F(\mathbf{x})}{\partial \mathbf{x}}\right)^T \hat{\mathbf{x}} \tag{3.7a}$$

$$\hat{\mathbf{x}}|_{t=t_R} = 0 \tag{3.7b}$$

$$\hat{\mathbf{x}}(t)|_\Gamma = 0 \tag{3.7c}$$

where t_R represents the final time of interest.

Eqs. (3.3) and (3.7) are the continuous forms of the TLM and adjoint model of MM5 if MM5 is symbolically written as (3.1). Unfortunately, such a procedure to derive the adjoint model, although good for simple models, is not easy to do for a complex model like MM5 for the following two reasons: (i) the partial integration procedure to derive (3.7) from (3.3) following (3.6) for a primitive equation model can be very tedious. With various physics options for which more than one expression is included, this becomes even more difficult. (ii) The discretization of the adjoint equation may become inconsistent with the original model and the accuracy of the gradient will be limited to the accuracy of the finite-difference scheme used in discretizing (3.7).

Another way of obtaining the adjoint model is to develop it directly from the discretized forward model, which will be illustrated as follows:

In discretized form, MM5 can be written in general as

$$\begin{aligned} \mathbf{x}(t_r) &= \mathbf{Q}_r(\mathbf{x})\mathbf{z} \\ \mathbf{z} &= (\mathbf{x}_0, \mathbf{y}_{t_1}, \mathbf{y}_{t_2}, \ldots, \mathbf{y}_{t_{ii}})^T \end{aligned} \tag{3.8}$$

where $\mathbf{z}$ represents the model input vector including both the IC $\mathbf{x}_0$ and boundary conditions $\mathbf{y}_{t_i}, i = 1, \ldots, ii$, and possibly model parameters if any. The discretized MM5 TLM can be directly developed from the discretized MM5 (3.8) and can be written as

$$\begin{aligned} \mathbf{x}'(t_r) &= \mathbf{P}_r(\mathbf{x})\mathbf{z}' \\ \mathbf{z}' &= (\mathbf{x}'_0, \mathbf{y}'_{t_1}, \mathbf{y}'_{t_2}, \ldots, \mathbf{y}'_{t_{ii}})^T \end{aligned} \tag{3.9}$$

where $\mathbf{P}_r = \partial\mathbf{Q}_r/\partial\mathbf{x}$. Therefore, the MM5 TLM can be obtained by a linearization procedure line by line.

The adjoint model is then defined as (see Section 2.3.2)

$$\hat{\mathbf{z}}^r = \mathbf{P}_r^T(\mathbf{x})\hat{\mathbf{x}}(t_r) \tag{3.10a}$$

$$\hat{\mathbf{x}}(t_r) = (forcing\ term), \quad r = R, R-1, \ldots, 0 \tag{3.10b}$$

under the Euclidean norm, where the hat represents adjoint variables, *forcing term*$(= \partial J/\partial\mathbf{x})$ depends on the forecast aspect (represented by a cost function J) one wishes to study (see section 2.2), and R is the total number of time levels at which forecast aspects are investigated. Comparing (3.9) with (3.10), we find that the adjoint model of MM5 is simply a transpose of the MM5 TLM. So the development of the MM5 adjoint model becomes a rewrite of the MM5 TLM model in a way which realizes the transpose of the original operations in the MM5 TLM. The gradient of a cost function J with respect to the control variable $\mathbf{z}$ obtained using (3.10) (see section 2.3.2 for a detailed derivation of gradient) is accurate to the machine accuracy. The detailed procedure of calculating the gradient of J using the MM5 adjoint model is provided later in Section 3.7.

This discretized method of building the adjoint model from a discrete forward model not only avoids the inconsistency generally arising from the derivation of the adjoint equations in analytic form followed by the discrete approximation, but also simplifies the procedure of constructing and debugging the adjoint model for a complex model. It can be further illustrated as follows: If we view the MM5 model operator as the result of the multiplication of a number of operator matrices:

$$\mathbf{Q}_r = \mathbf{Q}_1(\mathbf{x}_{r_1})\mathbf{Q}_2(\mathbf{x}_{r_2})\cdots\mathbf{Q}_N(\mathbf{x}_{r_N}), \tag{3.11}$$

where each matrix $\mathbf{Q}_i (i = 1, \cdots, N)$ represents either a subroutine or a single *DO* loop and $\mathbf{x}_{r_i}$ is a vector of variables depending on model predictive variables, then the TLM operator can be written as

$$\mathbf{P}_r = \mathbf{A}_1\mathbf{A}_2\cdots\mathbf{A}_N, \tag{3.12}$$

where matrix $\mathbf{A}_i(= \partial\mathbf{Q}_i/\partial\mathbf{x}_{r_i}), i = 1, \cdots, N)$ represents the linearized operation of $\mathbf{Q}_i$. The adjoint model operator $\mathbf{P}_r^T$ is then a product of sub-adjoint problems

$$\mathbf{P}_r^T = \mathbf{A}_N^T\mathbf{A}_{N-1}^T\cdots\mathbf{A}_1^T. \tag{3.13}$$

In this way, the discrete adjoint model can be constructed piece by piece. The discrete operations in the forward model have unique corresponding discrete operations in the adjoint model. The above derivation parallels the coding of the model where the algebraic operations are carried out by computer instructions (see Section 3.6). The correctness check for both the TLM and the adjoint model can also be carried out piece by piece. This is extremely handy and convenient, especially for dealing with various physical parameterization schemes. The tangent linear and adjoint versions for a specific scheme can be developed and tested separately from other parts of the model.

In general, for each physics routine *sub_name* in MM5, we developed three corresponding routines: *lsub_name* — linear version, *asub_name* — adjoint version, and *msub_name* — basic state version. For a linear subroutine, of course, *lsub_name* is not needed. For *lsub_name* and *asub_name* we use the same naming convention for the perturbation and adjoint variables respectively as the corresponding variables in the original code *sub_name*. In *msub_name*, only the input and output variables are changed into a different naming convention, by appending a number 9 to its original name. The naming convention for local variables in *msub_name* is not changed. *msub_name* does not exist if all the input and output variables in *sub_name* happen to be in the argument list or there is no future relevant nonlinear calculation following the CALL to this subroutine in the entire forward model.

3.2 Lateral boundary condition

The LBC used in MM5 includes values of all model variables and their time tendency at the five outermost points of the domain every y hours (update frequency of the large-scale boundary conditions). A relaxation method is used to "nudge" the model predicted variables toward the large-scale analysis near the lateral boundaries. The method includes

Newtonian and diffusion terms:

$$\frac{\partial \alpha}{\partial t} = \frac{5-n}{3}\frac{1}{10\Delta t}(\alpha_{LS} - \alpha_{MC}) - \frac{5-n}{3}\frac{\Delta s^2}{50\Delta t}(\alpha_{LS} - \alpha_{MC}). \qquad n = 2,3,4 \qquad (3.14)$$

where Δs is the grid spacing, α represents any model variables except the vertical velocity and cloud microphysics variables which is not nudged, MC denotes the model calculated quantities and LS the large-scale boundary conditions.

In the original setup of the MM5 model, the LBC includes values of all model variables and their time tendency at the five outermost points of the domain. However, the time tendencies of LBC are obtained from the values of model variables near the lateral boundaries several hours apart by a linear interpolation procedure. Therefore, either the tendencies or the model variables near the lateral boundaries should be considered as part of the control variables besides model IC. The actual MM5 input control variables are u, v, T, q_v, q_c, q_r, and p' at all model levels, w at inner model levels (see the next section), and the tendencies of all model variables at the five outermost points of the domain at every boundary updating time. These are summarized as $\mathbf{y}$ in (3.1) to represent the model input lateral boundary conditions. Currently, the time tendencies of model variables near the lateral boundaries are chosen as the control variables of LBC.

3.3 Upper and lower boundary conditions

Both options of the upper boundary conditions in the nonhydrostatic MM5 model are available in the MM5 adjoint model: (i) the fixed upper boundary and (ii) the upper radiative boundary condition. The vertical velocity at the surface is derived from the surface wind field, along with terrain information. Therefore, the vertical velocity w at the top and bottom model levels, originally included as part of the MM5 initial condition, should not be considered as part of the control variables. The part of the calculation for w at the top and bottom model levels, which were carried out in the preprocessing stage of the MM5 system, is thus added to the beginning of the MM5 forward model subroutine.

3.4 Adjoint physics

Adjoints of physical parameterization processes are required in the 4D-VAR system to increase the realism of the numerical model and to be able to assimilate new types of indirect observations which have a strong relation to moist physics and surface processes.

The MM5 adjoint model includes the following physical parameterizations:

1. Horizontal diffusion;
2. Vertical diffusion.
3. Dry convective adjustment;
4. Bulk-aerodynamic surface flux parameterization;
5. Resolvable scale precipitation processes;
6. Explicit treatment of cloudwater, rainwater, snow and ice;
7. The Kuo cumulus parameterization scheme;
8. The Grell cumulus parameterization scheme;
9. Surface energy equation;
10. Atmospheric radiation parameterization; and
11. Blackadar high-resolution model.

The adjoint version of each parameterization scheme is developed separately. The dry version of the MM5 adjoint model, which will be released in 1997, includes the simple physics (1)-(4), on top of the model dynamics.

Most of the problems encountered in the development of the adjoint physics are: the treatment of the "on-off" switches, the GOTO statements, the implicit redundant calculation, the loss of accuracy, the appearance of near-zero denominator, and the complex logical-related trace of input and output variables. In order to simplify the adjoint coding, we always start with replacing the GOTO statement with IF statements if possible. The "on-off" switches are retained, the physics TLM was first developed by linearizing the nonlinear model around the basic state at every time step while keeping the "on-off" switches the same as in the nonlinear model, and the adjoint physics is then obtained by transposing the operator of the physics TLM at the coding level.

3.5 TLM and adjoint code construction and model structure

When working with a numerical model which was developed without the adjoint concept, the following procedure must be followed to develop its adjoint model:

1. Identify all the independent control variables of the model;

   As mentioned in section 3.1, not all the original input variables can be viewed as the model's independent input variables. Also, the model's input constant array shall be identified from model variables since the linearization depends very much on this information.

2. Develop the TLM;

   The TLM is a linear version of the original nonlinear model and its solution represents a first-order approximation of the difference between perturbed and unperturbed nonlinear model solutions. It is obtained by linearizing the original model around a nonlinear model's solution in time — basic state. A general rule and several specific problems that shall be handled properly in the TLM development is presented in Chapter 4.

3. Test the correctness of the TLM;

   Eq. (3.15) (next page) shall be verified for all the model prognostic variables at any model grid point or for the whole domain. If the denominator equals zero, the values of both the numerator and denominator shall be examined separately.

4. Develop the adjoint model;

   Having a correct TLM, the adjoint model is developed based on the TLM. The adjoint model operator is simply the transpose of the TLM operator. Since we do not have an explicit TLM operator matrix, the transpose of the TLM has to be realized at the coding level. The MM5 adjoint model is obtained by writing the adjoint of each line in the TLM from the bottom to the top, while the calculation of those basic-state dependent coefficients in the adjoint model shall be kept the same as in TLM. Various rules of writing adjoint code and numerous examples illustrating these rules can be found in Chapter 4.

   Figure 3.1 shows the three flow charts of the main subroutines of the MM5 forward model, its TLM, and the MM5 adjoint model, with explicit interaction between MM5 and its TLM, or MM5 and its adjoint model, as well as indications for user interaction of gradient calculation of a specific cost function J^o depending on model predictions during the time window $[t_0, t_N]$. Given an IC $\mathbf{x}_0$, the MM5 model is integrated forward. The model prediction at every time step, $\mathbf{x}(t_r)$, is saved which is used as input of both the TLM and adjoint model. The backward integration of the adjoint model starts from a zero condition at the ending time of the assimilation window t_N: $\hat{\mathbf{x}}(t_N) = 0$. The model state $\mathbf{x}(t_r)$ consisting of wind, temperature, specific humidity, cloudwater, rainwater, and pressure perturbation are used to derive all the nonlinear coefficients in the TLM and adjoint model. This renders the computational cost of the TLM more than twice as expensive as the MM5 forward model integration itself due to the nonlinear calculation plus the linearization. The adjoint model integration takes even more time than that of TLM due to the repeated nonlinear calculations resulting from the fact that the adjoint calculation (backward) and the basic state calculation (forward) go in opposite directions. However, in this way less internal or disk memory is required than if we save more nonlinear coefficients.

3.6 Correctness check of TLM and adjoint model

The correctness of the TLM can be checked against the forward nonlinear model through the following formula:

$$\Phi(\alpha) \equiv \frac{\|\mathbf{Q}_r(\mathbf{z} + \alpha\mathbf{h}) - \mathbf{Q}_r(\mathbf{z})\|}{\|\alpha\mathbf{P}_r\mathbf{h}\|} = 1 + O(\alpha) \tag{3.15}$$

where $\mathbf{z}$ is a vector of all the input variables of the operator $\mathbf{Q}_r$. The values of $\Phi(\alpha)$ shall linearly approach unit value with increasing accuracy as α becomes progressively smaller. The initial perturbation with $\alpha\mathbf{h}$ in this range is "small" enough to ignore the higher-order terms and "large" enough to avoid machine round-off errors.

The correctness of the adjoint model can be checked by the following algebraic expression:

$$(\mathbf{P}_r\mathbf{z})^T(\mathbf{P}_r\mathbf{z}) = \mathbf{z}^T\left(\mathbf{P}_r^T(\mathbf{P}_r\mathbf{z})\right). \tag{3.16}$$

As we mentioned before, (3.15)-(3.16) can be used for checking the correctness of tangent linear and adjoint correspondences for any part of the code, a single *DO* loop or a subroutine or a combination of a few of these operations.

Adjoint check procedure:

1. Generate values of the input variables $\mathbf{z}$

   If the code is correct, (3.16) should be verified for all input values of $\mathbf{z}$. One can give $\mathbf{z}$ an artificial value. However, if $\mathbf{P}_r$ contains part of the physical processes, it's very difficult to produce a reasonable input value for $\mathbf{z}$ in order to activate those IF statements in the original code. If the input value of $\mathbf{z}$ doesn't result in a complete route for all parts of the code, then the check is incomplete. Therefore, in order to check the correctness of the adjoint code of a physical process, one may need to run the full model till the call to that physical process and save the input variables $\mathbf{z}$ and inputting constants for multiple checks that may follow.

2. Run the TLM code $\mathbf{A}$ and obtain the output vector of $\mathbf{y} = \mathbf{A}\mathbf{q}$. Calculate the left-hand-side value of (3.16): $wleft = \mathbf{y}^T\mathbf{y}$.

3. Use the output of the TLM code $\mathbf{y}$ as the input of the adjoint part $\mathbf{A}^T$ calculation ($\hat{\mathbf{y}} = \mathbf{y}$) and obtain the output the adjoint calculation $\hat{\mathbf{x}} = \mathbf{A}^T\hat{\mathbf{y}}$. Calculate the right-hand-side value of (3.16): $wright = \mathbf{q}^T\hat{\mathbf{x}}$.

4. Compare the values of $wleft$ and $wright$ to see whether they are equal to the machine accuracy. On CRAY-YMP, 13 digit accuracy is the best result to be expected. In some rare cases, less than 13 digit accuracy can still represent a none-error adjoint code. Our general experiences are that a minimum of 9 digit accuracy are required for a single-precision CRAY machine.

Another thing worth mentioning for a rigorous check of the adjoint code is the balancing problem, which shall be described as follows: Say that the output of the tangent linear operator $\mathbf{A}$, $\mathbf{y}$, contains two different variables: $\mathbf{y} = (\mathbf{y_1}, \mathbf{y_2})^T$, and the norm of $\mathbf{y_1}$ is much larger than the norm of $\mathbf{y_2}$ for a given input testing data set: $\mathbf{x}$. The adjoint test using (3.16) might still give a right test even if there is an error in the adjoint code $\mathbf{A}^T$ associated with the adjoint input $\mathbf{y_2}$, since the value of $\mathbf{y_1}^T\mathbf{y_1}$ is much larger than the value of $\mathbf{y_2}^T\mathbf{y_2}$ and the summation of $\mathbf{y_1}^T\mathbf{y_1} + \mathbf{y_2}^T\mathbf{y_2} = \mathbf{y}^T\mathbf{y}$ will not reflect such errors. In order to avoid the existence of such a hidden error in the adjoint code, one can either use a modified norm with proper scaling of the variables or use a more straightforward method: Check each variable $\mathbf{y_1}$ and $\mathbf{y_2}$ individually by repeating the check (3.16) twice by forcing $\mathbf{y_1} = 0$ and $\mathbf{y_2} = 0$ respectively before entering the adjoint operator $\mathbf{A}^T$.

3.7 Gradient calculation and its accuracy test

It is shown in Fig. 3.1, the gradient of a specific cost function J^o, depending on model predictions during the time window $[t_0, t_N]$, with respect to an IC $\mathbf{x}_0$ can be obtained as follows: integrate the MM5 model from $\mathbf{x}_0$ at time t_0. Save the model prediction at every time step, $\mathbf{x}(t_r)$, and save the forcing $\partial J^o/\partial \mathbf{x}(t_r)$ at those time steps when observations are available while the value of the cost function J^o is being calculated. Values of both saved nonlinear model trajectory $\mathbf{x}(t_r)$ and cost function related forcing $\partial J^o/\partial \mathbf{x}(t_r)$ are inputs to the adjoint model backward integration, which starts from a zero condition at the ending time of the assimilation window t_N: $\hat{\mathbf{x}}(t_N) = 0$. As mentioned before the nonlinear predictions $\mathbf{x}(t_r)$ are used to derive all the nonlinear coefficients in the adjoint model and $\partial J^o/\partial \mathbf{x}(t_r)$ are added to the adjoint variable $\hat{\mathbf{x}}$ at time t_r. The resulting value of the adjoint variables at the beginning time of assimilation window t_0 contains the value of the gradient of J^o with respect to the IC $\hat{\mathbf{x}}_0$, i. e., $\nabla_{\mathbf{x}_0} J^o = \hat{\mathbf{x}}_0$.

The correctness of the gradient calculation can be obtained through a procedure similar to the check of TLM (3.15):

$$\psi(\alpha) = \frac{J^o(\mathbf{x}_0 + \alpha\mathbf{h}) - J^o(\mathbf{X})}{\alpha\mathbf{h}^T \nabla J^o(\mathbf{x}_0)} = 1 + O(\alpha), \tag{3.17}$$

i. e., the values of $\psi(\alpha)$ shall linearly approach unit value with increasing accuracy as the order of the magnitude of α is decreased gradually.

When the cost function J^o consists of several terms (say $J^o = \sum_{i=1}^{II} J_i^o$), a more strict gradient check is to apply (3.17) to J_i^o separately, i. e., the gradients of J_i^o, $\nabla_{\mathbf{x}_0} J_i^o$ with respect to the IC $\mathbf{x}_0$ should be checked individually by setting J_i^o $(i \neq ii)$ equal to zero.

3.8 Minimization

The objective of 4D-VAR is to find an "optimal" initial condition (model parameter) that will result in a model solution which best fits background information (J^b), summarizing all the previous observational and forecast information, and various current observations distributed within a certain time and space interval (J^o), with proper error statistics for both the model and the observations. The "optimal" initial condition (model parameter) is sought based on the minimization of the objective function $J = J^b + J^o$. Both the large-scale limited-memory and truncated Newton minimization algorithms, mostly used for solving such a problem, require the gradient (∇J) information. One of the major classes of algorithms commonly used for large problems in meteorology is the limited-memory quasi-Newton method (Navon et al., 1992, Thépaut et al., 1991). From several $(m+1)$ gradient vectors $\mathbf{g}_i, i = k, k-1, \ldots, k-m$, the limited-memory quasi-Newton method forms an approximate to the search direction $\mathbf{d}_k$, which is defined as: $\mathbf{d}_k = -\mathbf{H}_k\mathbf{g}_k$, where $\mathbf{H}_k$ is the inverse Hessian matrix (the second derivative of the cost function with respect to the control variables: model initial condition and/or model parameter), the subscript k represents the number of iterations and m is an integer between 5 and 11 (Zou et al., 1993a). The vector $\mathbf{d}_k = -\mathbf{H}_k\mathbf{g}_k$ is a descent direction along which a step size α_k is found which satisfies:

$$J(\mathbf{x}_0^{(k)} + \alpha_k \mathbf{d}_k) = min_\alpha J(\mathbf{x}_0^{(k)} + \alpha \mathbf{d}_k) \tag{3.18}$$

Once the step-size α_k is determined, the initial condition obtained at the kth iteration is updated as:

$$\mathbf{x}_0^{(k+1)} = \mathbf{x}_0^{(k)} + \alpha_k \mathbf{d}_k \tag{3.19}$$

The next iteration continues by finding the next search direction $\mathbf{d}_{k+1}$ according to the new gradient information and so on.

Various limited-memory quasi-Newton methods (Shanno and Phua, 1980; Gill and Murray, 1979; Liu and Nocedal, 1989; and Buckley and Lenir, 1983) differ in the selection of m for $\mathbf{g}_i, i = k, k-1, \ldots, k-m$, the choice of the zeroth iteration Hessian matrix $\mathbf{H}_0$ (which is generally taken to be the identity matrix or some diagonal matrix for preconditioning purposes), the method for computing $\mathbf{H}_k\mathbf{g}_k$, and the line-search implementation which determines the suboptimal step size α_k (see (3.18)). If a standard minimization software is chosen, the only components needed from users are the values of the cost function J and its gradient ∇J at every updated control variable $\mathbf{x}_0^{(k)}$. The former can be obtained by integrating the nonlinear model forward and the latter by integrating the adjoint model backward with proper forcing terms added to the left-hand-side of the adjoint model.

We use the L-BFGS method of Liu and Nocedal (1989), which is described in section 2.3 of Zou et al. (1993a).

Figure 3.2 shows the flow chart of the main minimization program. The main program is separated into two parts:

(1) Read in IC and LBC, set all the ingredients in the cost function J including observations and weighting matrix, define units for the guess IC, perturbation IC, basic state, forcings, gradients and search direction, and form a simple scaling; and

(2) Carry out minimization procedure, which consists of the calculations of (i) the values of $J(\mathbf{x}_0^{(k)})$ and $\nabla_{\mathbf{x}_0^{(k)}} J$, (ii) the search direction $\mathbf{d}_k$, and (iii) the step size α_k, and print out the number of iterations, the number of function calls, the values of $J(\mathbf{x}_0^{(k)})$, $\nabla_{\mathbf{x}_0^{(k)}} J$, and α_k at each iteration.

3.9 Restart of minimization

The minimization procedure in 4D-VAR was carried out in an iterative fashion as already described in Section 3.8. The limited-memory quasi-Newton method of Liu and Nocedal (1989), which is implemented in the MM5 4D-VAR experiments, calculates the search direction as follows:

$$\begin{aligned}
\mathbf{d}^{(0)} &= -\,\mathbf{H}^{(0)}\mathbf{g}_0, \quad \mathbf{H}^{(0)} = \mathbf{I}, \\
\mathbf{d}^{(k)} &= -\,\mathbf{H}^{(k)}\mathbf{g}_k, \\
&\mathbf{H}^{(k)} = \mathbf{H}^{(k)}(\mathbf{H}^{(0)}, \mathbf{g}_k, \mathbf{g}_{k-1}, \ldots, \mathbf{g}_{k-m}, \mathbf{x}_0^{(k)}, \mathbf{x}_0^{(k-1)}, \ldots, \mathbf{x}_0^{(k-m)})
\end{aligned} \tag{3.20}$$

where $m = \min\{k, 4\}$. Therefore, the L-BFGS method generates the search direction (defined as the inverse Hessian matrix multiplied by the gradient) by using information from the last m (which is set as 5 in the code) quasi-Newton iterations. It thus requires $2Nm$ storage locations to save the vectors of gradient and state vector at the m most current iterations. For detailed expression of $\mathbf{H}^{(k)}$, see Zou et al. (1993a).

Examining the above updating formula for the search direction in the minimization procedure, we found that all the m most current values of the gradient and initial conditions: $\mathbf{g}_k, \mathbf{g}_{k-1}, \ldots, \mathbf{g}_{k-m}$, $\mathbf{x}_0^{(k)}, \mathbf{x}_0^{(k-1)}, \ldots, \mathbf{x}_0^{(k-m)}$ need to be saved properly, along with the iteration number, for the restart of minimization to produce an identical result without restart. This was done in the MM5 4D-VAR program.

3.10 Handling of disk space for large problem

The problem to be addressed in this subsection is the explosive growth in on-line computer memory for computing the gradient of a chosen cost function by the forward and backward integrations of the MM5 and its adjoint model, which characterize large-scale assimilation or sensitivity studies. The storage problem imposes a severe limitation on the size of adjoint studies, even on the largest computers. A strategy can be constructed that enables the forward MM5, the forward TLM, and the backward adjoint model runs to be performed in such a way that the on-line storage requirements can be traded for remote storage requirements, the mass storage system supported at NCAR.

Recall that both the TLM and adjoint model of MM5 are linear models linearized around the instantaneous nonlinear model solution. The MM5 model forecast of $\mathbf{x}(t_r) = (u(t_r), v(t_r), T(t_r), q(t_r), w(t_r), p'(t_r), q_c(t_r), q_r(t_r))^T$ is saved at every time step and inputted to the MM5 TLM and adjoint model. The intermediate basic-state dependent coefficients needed in the TLM and adjoint model are recalculated from the saved MM5 forecast variable $\mathbf{x}(t_r)$. The adjoint model requires the saved nonlinear model solution to be read in an opposite direction. In order to save the CPU time spent on reading in the basic state for the MM5 adjoint model, a direct access format is used for the basic state output. We then break the total assimilation period $[t_0, t_R]$ into several time intervals: $[t_0, t_{r_1}]$, $[t_{r_1+1}, t_{2r_1}]$, ..., and $[t_{Lr_1+1}, t_R]$. The value of r_1 depends on the disk quota limit allowed for the job. If the maximum file size for basic state saving is M gigabytes, then the maximum total time steps of which the basic-state can be saved on-line simultaneously is $r_1 = M/(8N)$, where N is the dimension of the model state $\mathbf{x}$. The total number of the times of the mass storage save, L, can then be determined as $L = R/r_1 + 1$. Therefore, after the r_1th time step, the basic state during the period of $[t_0, t_{r_1}]$ will be moved to the remote storage to free up the disk space for the continuing forecast in the time window $[t_{r_1} + 1, t_{2r_1}]$ and so on. When we integrate the adjoint model backward, the basic state during $[t_{Lr_1+1}, t_R]$ is used for the backward adjoint model integration from t_R to t_{Lr_1}. It is then removed and the basic state during $[t_{(L-1)r_1+1}, t_{Lr_1}]$ is read from the remote storage, and the adjoint model integration continues.

This generally-applicable strategy enables data assimilation and any other studies using MM5 TLM or adjoint model to be conducted on a significantly larger domain and/or with much higher resolution than would otherwise be possible, given the particular hardware constraints, and without compromising the outcome in any way. The computational time may increase depending on the speed of the data transfer process between the local machine and the remote storage.

3.11 Infrequent basic-state update

The MM5 TLM and adjoint models can be used for many applications. Since the TLM is a model linearized around the nonlinear model solution in time, the MM5 model solution at every time step is usually saved and then inputted to both the TLM and the adjoint model. This may present a storage problem for a large or even a medium-size job on current computers. For example, for a 6-h sensitivity calculation of a case with a grid-size of 62x79x27 and a time step of 1 min., the basic state saved from MM5 will need 6.07 GB disk space. If one wishes to extend the sensitivity study to 12 h, it can be done only if there is a way, either to shift some of the nonlinear model output to other places such as to the mass storage during the job execution time, or to use an infrequent update of the basic state for TLM and the adjoint model. One way to resolve this problem is to do an interactive shifting of the basic-state data onto some other place, as described in Section 3.10. Another way to reduce the large disk storage requirement of the basic state is to use an infrequent basic-state update. The former produces no approximation to the numerical results, though the computation may last much longer due to the data shifting processes, and the latter results in an approximation to the TLM solution and the gradient calculation. A careful study on the proper length of the basic state update frequency for the case of interest has to be conducted to ensure obtaining the convergence of the minimization and a reasonably good sensitivity result.

For those runs without a treatment to suppress gravity wave oscillations, delaying the infrequent update is suggested due to the presence of gravity oscillations in the basic state at the first few hours of model prediction.

There are three options for obtaining the approximated basic state at those time steps when basic states are not saved: a step-function approximation, a linear interpolation, and a bi-parabolic interpolation scheme. The linear interpolation scheme seems to be the best choice since it is much more accurate than the step-function approach, yet much cheaper than the bi-parabolic interpolation scheme while producing a similar results as the bi-parabolic interpolation scheme (Zou et al., 1997).

3.12 Adjoint sensitivity calculation

Sensitivity analysis estimates the impact of different perturbations in model inputs on the model forecast. For problems involving many input alterations (as MM5 does) and comparatively few model forecasted aspects (responses), the adjoint sensitivity formalism is computationally far more economical (Hall and Cacuci, 1983; Errico, 1993; Zou et al., 1993b) than any other methods. The sensitivity of one response to all the model parameters and the model initial condition and lateral boundary condition can be evaluated in terms of a single adjoint solution. In general, the gradient of the response function itself is used to assess the sensitivity of the forecast aspect without a specific concern for a specified perturbation to the model input variable. For instance, if the forecast aspect of interest is an implicit function of the model prediction at some time t_1, $\mathbf{x}(t_1)$ $(t_1 > t_0)$:

$$J(\mathbf{x}_0) = G(\mathbf{H}(\mathbf{x}(t_1))), \qquad (\mathbf{H} \text{ is a forward model operator}) \tag{3.21}$$

then the gradient of J with respect to the IC $\mathbf{x}_0$, the sensitivity, can be derived as

$$\nabla_{\mathbf{x}_0} J = \mathbf{P}^T_{t_1}(\mathbf{x}) \left(\frac{\partial \mathbf{H}}{\partial \mathbf{x}(t_1)} \right)^T \frac{\partial G}{\partial \mathbf{H}(\mathbf{x}_1)} \tag{3.22}$$

i.e., the gradient of J with respect to the IC can be obtained by integrating the adjoint model from time t_1 with the "initial" condition $\partial G / \partial \mathbf{x}_1$ backward in time. The resulting value of the adjoint variables at time t_0 is the gradient of J with respect to the IC $\mathbf{x}_0$. Notice that the adjoint model operator $\mathbf{P}^T$ is a function of the nonlinear model state $\mathbf{x}$ which is the MM5 forward model prediction starting from the IC $\mathbf{x}_0$ at the initial time t_0.

If the forecast aspect to be studied contains more than one time level model prediction and two different forward model operators $\mathbf{H}_1$ and $\mathbf{H}_2$, for instance,

$$J(\mathbf{x}_0) = G(\mathbf{H}_1(\mathbf{x}(t_1)), \mathbf{H}_2(\mathbf{x}(t_2))), \tag{3.23}$$

the gradient of J can be expressed as

$$\nabla_{\mathbf{x}_0} J = \mathbf{P}^T_{t_1}(\mathbf{x}) \left(\frac{\partial \mathbf{H}_1}{\partial \mathbf{x}(t_1)} \right)^T \frac{\partial G}{\partial \mathbf{H}_1(\mathbf{x}_1)} + \mathbf{P}^T_{t_2}(\mathbf{x}) \left(\frac{\partial \mathbf{H}_2}{\partial \mathbf{x}(t_2)} \right)^T \frac{\partial G}{\partial \mathbf{H}_2(\mathbf{x}_2)} \tag{3.24}$$

Due to the linear property of the adjoint model with respect to the adjoint variables, the gradient of J can be obtained by one single backward adjoint model integration starting from the time $t_m = \max\{t_1, t_2\}$ with

$$\hat{\mathbf{x}}(t) = \left(\frac{\partial \mathbf{H}_2}{\partial \mathbf{x}(t_2)} \right)^T \frac{\partial G}{\partial \mathbf{H}_2(\mathbf{x}_2)} \tag{3.25}$$

(assuming that $t_2 > t_1$) as the starting value of the backward adjoint model integration at the time t_m. When reaching the time t_1, the second forcing is added to the adjoint variables:

$$\hat{\mathbf{x}}(t_1) = \hat{\mathbf{x}}(t_1) + \left(\frac{\partial \mathbf{H}_1}{\partial \mathbf{x}(t_1)}\right)^T \frac{\partial G}{\partial \mathbf{H}_1(\mathbf{x}_1)}. \tag{3.26}$$

The resulting value of the adjoint variables at the time t_0 is the gradient of J consisting of the two terms in (3.24).

The distribution of the gradient could provide an indication concerning the most sensitive regions and the most sensitive variables.

3.13 4D-VAR experiment

In 4D-VAR, the adjoint model is used for the same purpose as in the sensitivity study: To obtain the gradient of a forecasted scalar function (cost function) with respect to model input vector $\mathbf{z}$. The cost function J measures the distance between the background information and initial guess, the model forecast and observation, weighted by the inverse of the background error covariance and the observational error covariance, respectively. Such a J should be defined by users to solve their own problems. Once J is defined, the values of J and its gradient with respect to the IC can be obtained following the procedure described in Section 3.10. Any large-scale unconstrained minimization can then be used to find the minimum of J, and $\mathbf{x}_0^*$, is usually called the optimal IC (see Section 3.8). A limited-memory quasi-Newton method is provided in the MM5 4D-VAR system.

A good 4D-VAR experiment depends on a proper definition of J. Examining (2.81), a complete 4D-VAR system has to include the following components:

(i) the estimation and specification of background term ($\mathbf{x}_b$ and $\mathbf{B}^{-1}$) and observation error statistics ($\mathbf{O}_r$);

(ii) the forecast model and its adjoint ($\mathbf{Q}_r$ and $\mathbf{P}_r^T$);

(iii) the forward observation operators ($\mathbf{H}_r$) and their corresponding adjoint operators ($\mathbf{H}_r^T$). $\mathbf{H}_r$ shall correctly calculate the model correspondence of the ith type of observations; and

(iv) the control of gravity wave oscillations.

At present, a proper background term, J^b, for MM5 is not yet available. However, a prototype 4D-VAR experiment can still be carried out by using either the NCEP (National Center for Environmental Prediction) or ECMWF analysis as a simplest background information.

Once the cost function J is properly defined, including the definition of the weighting function, a scaling factor must be defined before the minimization procedure can be executed. Scaling is a crucial issue in the success of unconstrained minimization problems (Gill et al., 1981). In meteorology, the variables in the control vector have enormously different magnitudes varying over a range of several orders of magnitude. Scaling by variable transformation converts the variables from units that reflect the physical nature of the problem to units that display desirable properties for the minimization process. A simple scaling is provided as an option in the MM5 4D-VAR system, which is similar to what was done in the paper by Navon et al., (1992) in section 3c.

3.14 A brief summary

A nonhydrostatic mesoscale adjoint model has been developed which is suitable for many synoptic and mesoscale studies for a wide variety of problems requiring adjoint techniques. The adjoint model has been developed and coded based on the Penn State/NCAR mesoscale model version 5 (MM5), and faithfully following the original MM5 code. It allows an easy future update of the MM5 adjoint model as MM5 is updated and changed. A backward in time integration of the MM5 adjoint model can produce an accurate gradient of any forecast aspect in a computationally efficient way. A proper handling of lateral boundary conditions is provided with the option to either perturb or fix them. The consequences of the optimal control, or of perturbing lateral boundary conditions on the simulated flow, can be considered.

Technical details involved in carrying out 4D-VAR and sensitivity analysis were described in this chapter. A few computational aspects of using the MM5 adjoint model were discussed. These include the capability of the restart of the minimization procedure, the possibility of trading the on-line storage requirements for remote storage requirements to eliminate the limitation of disk space for large problems without compromising the results in any way, and the option of using an infrequent basic-state update. A number of other applications of the MM5 TLM and adjoint model, including the treatment of model errors, the use of an incremental approach, the inverse TLM integration, and the singular vector calculation with and without physics, were described in the previous chapter for a general numerical prediction model. They expand the usefulness of the MM5 TLM and the adjoint model.

The most difficult problems remaining are the handling of the interaction of multiple meshes, and a proper specification of the background error and accounting for model errors of MM5, which require further substantial theoretical study and an experimental program.

CHAPTER 4: PRACTICAL ADJOINT CODING IN METEOROLOGY

The progressively wide use of the adjoint models in many applications has provided motivation for summarizing our experiences in developing various adjoint models, ranging from spectral to finite-difference, and simple to complicated numerical models with various physical parameterization schemes.

While an automatic adjoint works for a "so-called" clean code, most adjoint model developments are carried out by hand, with a "dirty" code at one's disposal. The practical situation of building an adjoint model from a pre-existing model (which is developed with no knowledge of adjoint) which is usually not "clean" and might contain many computational approximations, makes the automatic adjoint coding not directly applicable to most problems. It is the intention of this section to highlight many problems which may appear in the practical coding of adjoint models.

Given a complex numerical model of weather phenomena, we require an adjoint model to calculate the gradient of some of the model outputs $\mathbf{y} \equiv \mathbf{Cx}(t)$ with respect to some or all of the inputs $\mathbf{x}_0$. The adjoint model can be produced either by hand or by automatic adjoint generator (Giering, 1996). In this appendix, we will summarize some hand-coding-adjoint experiences, which shall be valuable for many newcomers to this field.

As mentioned in Chapter 3, the MM5 adjoint model is developed based on the MM5 TLM model. In this chapter, we will summarize some of the experiences obtained in developing the MM5 adjoint model system with full physics. We have also included a few examples we encountered in developing the adjoint model of the NCEP global operational spectral model with various physical processes.

4.1 TLM coding — linearization

Following are a few items useful for the model linearization:

(i) Identify all the input and output variables for the entire model, each subroutine, and every do loop.

(ii) Linearize nonlinear terms with respect to those variables identified in (i). Most of the nonlinear terms in a numerical model consist of arithmetic operators (e.g., addition,

Preceding page blank

multiplication, division) or intrinsic functions (sin, exp, $(\ )^a$, etc.) which are rather straightforward to linearize.

(iii) Pay special attention to those terms in which a numerator in the nonlinear model becomes a denominator in the TLM (such as with SQRT function). Add a small number ϵ, a number close to machine accuracy, to the denominator which may become zero value.

(v) Check if there exists an implicit solver using an iterative method and treat it differently.

The easiest way to construct the tangent linear code is to write two lines in the tangent linear version for each line in the nonlinear model, with the first line calculating the perturbation term and the second line calculating the basic state determined coefficients. Calculating the perturbation part before the calculation of the basic state part may become a must in certain cases (see example 2). The basic-state calculation is not necessary if it is not associated with the nonlinear coefficients.

Following are examples which describe each of the above items that could occur in the practical linearization procedure:

4.1.1 Inputs and Outputs

The first and most important thing for the TLM and adjoint model development is to correctly determine the input variables, input constants, and output variables for any part of the code. The following example is used to illustrate this point:

Example 1:

```
A=B*C
D=A*C+A
```

Knowing that both B and C are inputs is not sufficient. One must decide whether they are an input variable or they are a constant before writing the corresponding tangent linear code. Those quantities which depend on model input control variables are variables, and others are constants. There are three possibilities to example 1:

(i) B is a constant, C is a variable. In this case, the tangent linear code of example 1 takes the following form:

```
A=B*C
A9=B*C9
D=C9*A+A9*C+A
D9=A9*C9+A9
```

(ii) B is a variable, C is a constant:

```
A=B*C
A9=B9*C
D=A*C+A
D9=A9*C+A9
```

(iii) Both B and C are variables:

```
A=B9*C+C9*B
A9=B9*C9
D=C9*A+A9*C+A
D9=A9*C9+A9
```

where variables appended with number 9 represent the basic state, which shall possess the same value as that in the nonlinear model solution without 9.

4.1.2 Linearization

Once the input and output variables are correctly determined, the next step is to linearize the code. If it is a linear code, copy it. For those nonlinear terms, linearization must be carried out properly. For the advection terms or any terms similar to them, linearization is rather straightforward.

Example 2:

```
A=B*EXP(0.7*C)
A=A*B
```

where both B and C are input variables.

The tangent linear code corresponding to this part is

```
A =B*EXP(0.7*C9)+B9*0.7*EXP(0.7*C9)*C
A9=B9*EXP(0.7*C9)
A=A*B9+A9*B
A9=A9*B9
```

For the first line of the code in example 2, we can either place the perturbation calculation before the basic state calculation, or vice versa. Both are correct. For the second line of the code in the above example, we must calculate the basic state after the perturbation line. Otherwise, the value of A9 in the third line of the TLM code is incorrect.

Another example we would like to mention here is an instance where a numerator becomes a denominator during a linearization procedure.

Example 3:

```
A = SQRT(B)
A = A**0.3
```

The tangent linear code is

```
A = 0.5*B/SQRT(B9)
A9= SQRT(B9)
A = 0.3*A/A9**0.7
A9= A9**0.3
```

It is not a problem if B=0 or A=0 in the nonlinear model. But in the TLM, B9=0 or A9=0 will make the TLM blow up. One way to avoid this is to add a small number ϵ close to machine accuracy to the denominators. The TLM code will take the form:

```
A = 0.5*B/(SQRT(B9)+ε)
A9= SQRT(B9)
A = 0.3*A/(A9**0.7+ε)
A9= A9**0.3
```

4.1.3 Dealing with iterative solver

However, there are other cases where linearization is not as straightforward as shown above. For example, in the cumulus convection subroutine of the NCEP model, the dew-point temperature T_d is calculated from the vapor pressure v_p via an iterative method. The tangent linear version of this calculation cannot be obtained by directly linearizing the nonlinear code, i.e., linearizing the whole procedure of the iteration. One must obtain the original formulation of the relation between T_d and v_p, derive the analytical tangent linear equation of the relevant equation, and write the independent linear code of this part using the same variable notations as that used in the nonlinear model. The linear check should be carefully done in this case to avoid any possible inconsistent usage of variables.

Example 4:

The dew point temperature T_d is calculated via a Newton iterative method using the following formula:

$$v_p = 0.611 \left(\frac{T_0}{T_d}\right)^A e^{(A+B)\left(1-\frac{T_0}{T_d}\right)} \tag{4.1}$$

where

$$A = \frac{c_l - c_v}{R_v}, \quad B = \frac{L_0}{R_v T_0}, \quad T_0 = 273.16^o K$$

where c_l is the specific heat of liquid water (which enters through the temperature dependence on L).

The value of T_d as a function of v_p is obtained by finding the root of $f(x) = 0$, where $f(x)$ is defined as

$$f(x) = A \ln x - (A+B)x - [\ln v_p - \ln 0.611 - (A+B)] \quad x = \frac{T_0}{T_d}. \tag{4.2}$$

The iteration starts from $x^{(0)} = 1.0$ and ends whenever the following convergence criterion is satisfied:

$$|x^{(n)} - x^{(n-1)}| \leq 10^{-8}, \tag{4.3}$$

where superscript (n) represents the n-th iteration.

Differentiating equation (A1) we have

$$\begin{aligned} v_p' &= 0.611 * \left[Ax^{A-1} - (A+B) * x^A\right] e^{(A+B)(1-x)} x', \\ x' &= -\frac{T_0}{T_d^2} T_d'. \end{aligned} \tag{4.4}$$

Therefore, the perturbation T_d' can be calculated if v_p' is known. The value of T_d and, thus x, appearing as the coefficient in (4.4) is obtained by the original iterative method solving (4.2).

Following is the corresponding nonlinear code:

```
SUBROUTINE ZDEWPT(VP,TD,N)
SAVE
REAL RA, RAPB, RB, RCH, RD, RDVP, RGS, RLOG3, RLVP, RN, RNT, RT
REAL RT3, RTEST, RVP, RVP1, RVP2, RVP3
DIMENSION TD(N),VP(N)
RT3= 2.7316E+2          (T0)
RVP3= 6.1078E+2 *1. E -3  (vp  at T0)
RLOG3=LOG(RVP3)
```

```
    RA=( 4.1855E+3 - 1.8460E+3 )/ 4.6150E+2 (A in eq.(4.1))
    RB= 2.5000E+6 /( 4.6150E+2 * 2.7316E+2 ) (B in eq.(4.1))
    RAPB=RA+RB
    RTEST=1.E-8  (criterion for convergence of newton iteration)
    RGS=1.E0   ($x^{(0)}$)
    DO 20 NN=1,N
       RVP=VP(NN)
       RLVP=LOG(RVP)-RLOG3-RAPB
10     RN=RA*LOG(RGS)-RAPB*RGS-RLVP  ($f(x)$, Newton iteration loop.)
       RD=(RA/RGS)-RAPB  (now get its derivative)
       RCH=RN/RD
       IF (ABS(RCH).LT.RTEST) GO TO 15  (the desired change in the guess)
       RGS=RGS-RCH  (need more iterations)
       GO TO 10
15     RT=RT3/RGS  ($T_0/T_d$)
       TD(NN)=RT
20  CONTINUE
    RETURN
    END
```

The tangent linear code corresponding to the above subroutine is

```
    SUBROUTINE LDEWPT(VP,TD,N,VP9,TD9)
    SAVE
    REAL RA, RAPB, RB, RCH, RD, RDVP, RGS, RLOG3, RLVP
    REAL RN, RNT,RT, RT3, RTEST, RVP, RVP1, RVP2, RVP3
    DIMENSION TD(N),VP(N),TD9(N),VP9(N)
    RT3= 2.7316E+2    (triple point)
    RVP3= 6.1078E+2 *1. E -3
    RA=( 4.1855E+3 - 1.8460E+3 )/ 4.6150E+2
    RB= 2.5000E+6 /( 4.6150E+2 * 2.7316E+2 )
```

```
      RAPB=RA+RB


      CALL ZDEWPT(VP9,TD9,N)
      DO 20 NN=1,N
         X9=RT3/TD9(NN)
         COE=RVP3*(RA*X9**(RA-1)-(RAPB)*X9**RA)*EXP(RAPB*(1-X9))
         X=VP(NN)/COE
         TD(NN)=-X*TD9(NN)*TD9(NN)/RT3
 20   CONTINUE
      RETURN
      END
```

From the above example, we do not find much similarity between the original nonlinear code and its linear code as compared to linearizing, for example, advection terms.

4.2 Adjoint model coding — transposition

Mathematically, the adjoint model is simply the transpose of the TLM. The implementation of the transposition of a large computer code in languages such as Fortran 77, without the matrix of TLM being explicitly available, is not easy. The transpose of the TLM has to be obtained by translating the TLM line by line into a sequence of computer code which realizes the transpose operation of TLM. This becomes even more complicated when the forward model is not clean, meaning the input-output variables for different parts of the model are not easily identifiable and the code contains some redundant calculations, GOTO statements, multiple usages of one variable, and cancellation of effective digits. Although the MM5 adjoint model with physics has already been developed, incorporating various types of observations into MM5 or conducting a sensitivity study of some indirect forecast aspects using the MM5 adjoint model requires the adjoint of the forward observation operators. Therefore, additional adjoint coding is still needed for various applications even if the MM5 adjoint model is made available.

Following is a list of rules for developing adjoint models:

1. Matrix method: A single loop, or several loops or a subroutine in the TLM can always be expressed as an operation of a matrix multiplied by a vector, i.e., $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{x}$ is a vector of all the input variables and $\mathbf{y}$ a vector of the output variables. The adjoint code corresponding to this piece of code shall always realize the operation of $\hat{\mathbf{x}} = \mathbf{A}^T\hat{\mathbf{y}}$. Note that if the input $\mathbf{x}$ is reused after the operation $\mathbf{y} = \mathbf{A}\mathbf{x}$ in the TLM, $\mathbf{x}$ should be included in the output vector $\mathbf{y}$.

2. Reverse rule: As indicated in (3.12) and (3.13), the adjoint version of any part of the TLM works in a reverse order, meaning the first operation in a subroutine of the TLM will be the last operation in its adjoint subroutine, the last will be the first, and so on. However, the basic state which provides the coefficients to both the TLM and adjoint model shall be calculated in a forward mode as was done in the nonlinear model MM5. The coefficients depending on the nonlinear model basic state solution must be correctly re-calculated in the adjoint models, particularly those variables which are constantly updated.

3. Check whether any input variable of $\mathbf{A}$ is reused or not at a later stage, which requires a global familiarity of the whole TLM.

4. Pay attention to the hidden reuse in a recursive loop, especially an implicit recursive loop.

5. Find places where some variables are re-defined.

6. Find redundant calculations: For those operations in which values of some variables are calculated but not used afterward in TLM, a cleanup must be made to remove these operations or to zero out these variables in the adjoint model.

7. Special caution must be paid to variables which are sometimes used as model constants and other times as variables.

8. Check whether an IF statement used variables as its decision components — treatment of "on-off" switches.

9. Replace GOTO statements with IF ... THEN ... ELSE.

10. Be aware of the cancellation of the effective digits in the TLM.

In the following we provide examples of the adjoint coding in order to better illustrate these rules.

4.2.1 Matrix method

The first and foremost important rule of the adjoint coding is the matrix method. This is the basic method of constructing an adjoint model from a TLM. One can always view a loop, or several loops, or a subroutine, or a model as an operation matrix $\mathbf{A}$ acting on a vector $\mathbf{x}$ (the input variables). The result is represented as an output vector $\mathbf{y}$:

$$\mathbf{y} = \mathbf{A}\mathbf{x} \tag{4.5}$$

The adjoint code development is to write the code corresponding to the following transposition:

$$\hat{\mathbf{x}} = \mathbf{A}^T\hat{\mathbf{y}} \tag{4.6}$$

where $\mathbf{A}^T$ represents the transpose operation of $\mathbf{A}$.

*Example 5: (*see Navon et al., 1992*)*

```
      DO 130 I=1,N-1
         X(I)=a*Y(I+1)
130   CONTINUE
```

Obviously Y is the input variable and X is the output variable. However, Y might be used again after this loop in the tangent linear model. If so, Y should also be considered as an output variable of this part of the code, i.e., there are two possibilities:

1. X is the only output variable, i.e., Y is not used after this loop in the forward model;
2. Both X and Y are output variables, i.e., Y is used as input variable again for the other part of the model after loop 130;

The **A** matrix thus takes two different forms. For the first case,

$$\mathbf{A}_1 = \begin{pmatrix} 0 & a & 0 & \cdots & 0 & 0 \\ 0 & 0 & a & \cdots & 0 & 0 \\ . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . \\ 0 & 0 & 0 & \cdots & 0 & a \end{pmatrix}_{(N-1)\times N},$$

and for the second case,

$$\mathbf{A}_2 = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 0 & a & 0 & \cdots & 0 & 0 \\ 0 & 0 & a & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \\ . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . \\ 0 & 0 & 0 & \cdots & 0 & a \end{pmatrix}_{(2N-1)\times N}.$$

Therefore, the adjoint code of loop 130 in Example 2 is either

$$\mathbf{y} = \mathbf{A}_1^T \mathbf{x} \quad \text{when X is the only output variable}$$

or

$$\mathbf{y} = \mathbf{A}_2^T \mathbf{x} \quad \text{when both X and Y are output variables}$$

depending whether or not the input variable **x** is reused or not.

The adjoint code corresponding to the loop 130 thus becomes:

```
      DO 130 I=1,N-1
         Y(I+1)=a*X(I)      (Y is not reused after loop 130 in TLM)
130   CONTINUE
```

or

```
      DO 130 I=1,N-1
         Y(I+1)=a*X(I)+Y(I+1)     (Y is reused after loop 130 in TLM)
130   CONTINUE
```

4.2.2 Reverse rule

Since the adjoint operation is simply the transpose of the tangent linear operation, the adjoint coding goes in the reverse order, i.e., if the TLM contains the following calculation:

$$\mathbf{x} = \mathbf{P}_1\mathbf{P}_2\mathbf{P}_3\mathbf{P}_4\mathbf{y},$$

the adjoint of the above calculation is

$$\mathbf{y} = \mathbf{P}_4^T\mathbf{P}_3^T\mathbf{P}_2^T\mathbf{P}_1^T\mathbf{x}.$$

Example 6:

For example, in the tangent linear model, we have a series of calculations:

$$\mathbf{x}_1 = \mathbf{P}_1\mathbf{y}_1$$

$$\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \mathbf{P}_2 \begin{pmatrix} \mathbf{y}_2 \\ \mathbf{x}_1 \end{pmatrix}$$

$$\mathbf{y}_1 = \mathbf{P}_3\mathbf{x}_1$$

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \mathbf{P}_4 \begin{pmatrix} \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}$$

The above calculation can be written as

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \tilde{\mathbf{P}}_4\tilde{\mathbf{P}}_3\tilde{\mathbf{P}}_2\tilde{\mathbf{P}}_1 \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix}$$

where

$$\tilde{\mathbf{P}}_1 = \begin{pmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{I} \\ \mathbf{P}_1 & 0 \end{pmatrix}, \tilde{\mathbf{P}}_2 = \begin{pmatrix} \mathbf{I} & 0 & 0 \\ 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 \end{pmatrix} \tilde{\mathbf{P}}_3 = \begin{pmatrix} \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{P}_3 & 0 \\ 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & \mathbf{I} \end{pmatrix}, \tilde{\mathbf{P}}_4 = (0 \quad \mathbf{P}_4)$$

Notice that one cannot simply write the full computation (1)-(4) as

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \mathbf{P}_4\mathbf{P}_3\mathbf{P}_2\mathbf{P}_1 \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix}.$$

The adjoint version of (1)-(4) is

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \tilde{\mathbf{P}}_1^T\tilde{\mathbf{P}}_2^T\tilde{\mathbf{P}}_3^T\tilde{\mathbf{P}}_4^T \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix},$$

i.e.,

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \tilde{\mathbf{P}}_4^T \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \tilde{\mathbf{P}}_3^T \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \tilde{\mathbf{P}}_2^T \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = \tilde{\mathbf{P}}_1^T \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}$$

4.2.3 Reused variables

In example 5 we briefly described the difference in the adjoint coding for reused or not reused variables. Here is another example which has an implicit reuse of a variable:

Example 7

Linear code:

```
      DO 30 K=1,KK
      DO 30 I=1,II
         Y(I,K)=0.
         DO 40 L=1,KK
            Y(I,K)=Y(I,K)+X(I,L)*FLOAT(K)
40       CONTINUE
30    CONTINUE
```

If we assume that the input variable of the above code is X and the output variable is Y, we mean that the variable X is not reused after loop 30.

The point we wish to emphasize here is that the input variable X(I,L) is reused implicitly in the loop 30 for different K (K=1,...,KK).

Therefore, the adjoint code should be written as:

```
      DO 20 K=1,KK
      DO 20 I=1,II
         X(I,K)=0.
20    CONTINUE
      DO 30 K=1,KK
      DO 30 I=1,II
      DO 30 L=1,KK
         X(I,L)=Y(I,K)*FLOAT(K)+X(I,L)
30    CONTINUE
```

4.2.4 Recursive

For a recursive loop (for example, vertical velocity calculation), the newly calculated value of the variables are reused in the same loop. Therefore, one recursive loop does not correspond to one matrix operator. It is the multiplication of N matrix, where N is the total number of the recursive index.

Example 8:

Linear code:

```
        X(1)=1.0
        DO 10 I=2,3
           X(I)=7.7*X(I-1)
10      CONTINUE
```

The above code is equivalent to the calculation

$$\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} = \mathbf{P}_2\mathbf{P}_1 \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix}$$

where

$$\mathbf{P}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 7.7 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{3\times3} , \quad \mathbf{P}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 7.7 & 0 \end{pmatrix}_{3\times3} .$$

Therefore, the adjoint code should realize the calculation represented by

$$\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} = \mathbf{P}_1^T\mathbf{P}_2^T \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix}$$

where

$$\mathbf{P}_2^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 7.7 \\ 0 & 0 & 0 \end{pmatrix}_{3\times3} , \quad \mathbf{P}_1 = \begin{pmatrix} 1 & 7.7 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}_{3\times3} .$$

The first operation:

$$\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} = \mathbf{P}_2^T \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix}$$

is equivalent to

X(2)=7.7*X(3)+X(2)

and the second one:

$$\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix} = \mathbf{P}_1^T \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix}$$

is equivalent to the code

X(1)=7.7*X(2)+X(1)

Therefore, the adjoint code should take the form

Adjoint code:

```
       DO 10 I=3,2,-1
          X(I-1)=7.7*X(I)+ X(I-1)
10    CONTINUE
```

i.e., the order of the loop on I must be reversed.

4.2.5 Redundancy

This problem arises when the forward model is not clean. For example, in MM5 SOLVE3 subroutine, we have

Example 9:

```
      JBNES=2
      DO 10 J=1,JJ
         A(J)=B(J)+W(J)
         IF (J.EQ.JBNES) THEN
            QDOT(JBNES-1)=a(J)*W(J-1)
         ENDIF
         QDOT(J)=b(J)*W(J)
```

```
10    CONTINUE
```

The calculation QDOT(J)=b(j)*W(J) for J=1 is overwritten by QDOT(JBNES-1)=a(j)*W(J-1) when J=2, i.e., QDOT(J)=b(J)*W(J) for J=1 is a redundant calculation. It is not wrong to be kept in the forward model or the TLM but it shall be taken care of in the adjoint coding:

```
      JBNEST=2
      DO 10 J=1,JJ
         IF (J.NE.1) THEN     ! This is very important to add here.
            W(J)=b(J)*QDOT(J)
         ENDIF
         IF (J.EQ.JBNES) THEN
            W(J-1)=a(J)*QDOT(JBNES-1)
         ENDIF
         B(J)=A(J)
         W(J)=A(J)
10    CONTINUE
```

4.2.6 Mixing used variables

Sometimes one array was used for different purposes. In the following example, for instance, the array TGB (ground temperature) was temporarily used as a different variable depending on pressure variable PPB. Special attention must be paid to this case.

Example 10: (bulk PBL)

```
      DO 300 I=2,ILX
         TGDSA(I)=TGB(I,J)    ( TGB is assumed to be a constant array)
         PS1 =PPB (I,J,KL)/PSB(I,J)/1000.
         PS19= PSB(I,J)+PTOP+PPB9(I,J,KL)/PSB(I,J)/1000.
         TGB(I,J)=-ROVCP*TGB(I,J)*(100./PS19)**(ROVCP-1)*100.*PS1/(PS19*PS19)
```

```
C        (TGB is being used as a tentative variable array)
         TGB9(I,J)=TGB(I,J)*(100./PS19)**ROVCP

300   CONTINUE
```

The adjoint of the above loop 300 has to be written as

```
      DO 300 I=2,ILX
         TGDSA(I)=TGB(I,J)    ! TGB is constant array
         PS19= PSB(I,J)+PTOP+PPB9(I,J,KL)/PSB(I,J)/1000.
         PS1=-ROVCP*TGDSA(I)*(100./PS19)**(ROVCP-1)*100.*TGB (I,J)/(PS19*PS19)
         PPB (I,J,KL)=PS1/PSB(I,J)/1000. +PPB (I,J,KL)
300    CONTINUE
```

It is wrong if we write the 4th line in adjoint loop 300 as

```
      PS1=-ROVCP*TGB(I,J)*(100./PS19)**(ROVCP-1)*100.*TGB (I,J)/(PS19*PS19)
```

There are also cases where one array was sometimes used as a working array and sometimes used as a variable. For example, in SOLVE3, SCR3 was used as a working array for subroutines (DIFFU, VADV) which calculate diffusion and vertical advection, it is also used as a model variable representing a PBL tendency of the temperature T in the subroutine BLKPBL. In order to avoid such confusion, SCR3 in DIFFU and VADV should be changed into another name of working array.

4.2.7 "on-off" processes

All the "on-off" switches in MM5 are kept in the MM5 adjoint model, and they are turned on or off depending on the basic state of the nonlinear model solution.

Example 11:

Nonlinear code:

```
      SCA = 0.0
      DO 20 K=1,KL
         SCA =SCA +QVTEN(I,K)*DSIGMA(K)
20    CONTINUE
      IF (SCA .LT. QDCRIT ) THEN
         DO 80 K=2,KL
            SCR3(I,K)=TWT(K,1)*QVA(I,J,K)+TWT(K,2)*QVA(I,J,K-1)
80       CONTINUE
      ENDIF
```

Linear code:

```
      SCA = 0.0
      SCA9 = 0.0
      DO 20 K=1,KL
         SCA =SCA +QVTEN(I,K)*DSIGMA(K)
         SCA9=SCA9+QVTEN9(I,K)*DSIGMA(K)
20    CONTINUE
      IF (SCA9 .LT. QDCRIT ) THEN
         DO 80 K=2,KL
            SCR3(I,K)=TWT(K,1)*QVA(I,J,K)+TWT(K,2)*QVA(I,J,K-1)
80       CONTINUE
      ENDIF
```

Adjoint code:

```
      SCA9 = 0.0
      DO 20 K=1,KL
         SCA =SCA +QVTEN(I,K)*DSIGMA(K)
         SCA9=SCA9+QVTEN9(I,K)*DSIGMA(K)
```

```
20    CONTINUE
      IF (SCA9 .LT. QDCRIT ) THEN
         DO 80 K=2,KL
            SCR3(I,K)=TWT(K,1)*QVA(I,J,K)+TWT(K,2)*QVA(I,J,K-1)
            QVA(I,J,K)=TWT(K,1)*SCR3(I,K)+QVA(I,J,K)
            QVA(I,J,K-1)=TWT(K,2)*SCR3(I,K)+QVA(I,J,K-1)
80       CONTINUE
      ENDIF
      DO 20 K=1,KL
         QVTEN(I,K)=SCA*DSIGMA(K)+QVTEN(I,K)
20    CONTINUE
```

Therefore, the switch depends only on the basic state SCA9, not the perturbation SCA.

4.2.8 Cancellation of the effective digits

In developing the adjoint of the Grell cumulus scheme, we had difficulty obtaining more than 8 digit accuracy for the adjoint check, which was found to be due to the cancellation of effective digits in the original nonlinear code.

Example 12:

Nonlinear code:

```
      MBDT=DTIME*5.E-03
      DO 175 I=ISTART,IEND
         IF (AA0(I).LT.0.) GO TO 175
         F= (AA1(I)-AA0(I))/DTIME
         XK=(XAA0(I)-AA0(I))/MBDT
         XMB(I)=-F/XK
         IF (F.LE.0.OR.XK.GE.0.) XMB(I)=0.
 175  CONTINUE
```

Linear code:

```
      MBDT=DTIME*5.E-03
      DO 175 I=ISTART,IEND
         IF (AA09(I).GE.0.) THEN
            F = (AA1(I)-AA0(I))/DTIME
            F9= (AA19(I)-AA09(I))/DTIME
            XK=(XAA0(I)-AA0(I))/MBDT
            XK9=(XAA09(I)-AA09(I))/MBDT
            IF (F9.LE.0..OR.XK9.GE.0.) then
               XMB(I)=0.
               XMB9(I)=0.
            ELSE
               XMB(I)=-F/XK9+(F9*XK)/(XK9*XK9)
               XMB9(I)=-F9/XK9
            ENDIF
         ENDIF
 175   CONTINUE
```

Adjoint code:

```
      DO 175 I=ISTART,IEND
         IF (AA09(I).GE.0.) THEN
            F9= (AA19(I)-AA09(I))/DTIME
            XK9=(XAA09(I)-AA09(I))/MBDT
            IF (F9.LE.0..OR.XK9.GE.0.) then
               XMB(I)=0.
               F=0.
               XK=0.
            ELSE
               F=-XMB(I)/XK9
               XK=F9*XMB(I)/(XK9*XK9)
            ENDIF
            XAAO(I)=XK/MBDT
            AAO(I)=-XK/MBDT
            AA1(I)=F/DTIME
            AAO(I)=-F+AAO(I)/DTIME
         ENDIF
 175  CONTINUE
```

Symbolically, both the tangent linear and adjoint codes are correct. But the accuracy check gives only 8 digits. We found that it results from the fact that the difference of the two very close numbers AA1(I) and AA0(I), XAA0(I) and AA0(I) are divided by two big numbers DTIME and MBDT, respectively. DTIME and MBDT had a common factor of DTIME. The results of these operations F and XK are divided afterward. If we modify the TLM and adjoint codes into the following equivalent operations:

Modified linear code:

```
      MBDT=DTIME*5.E-03
      CMBDT = 5.E-03
      DO 175 I=ISTART,IEND
         IF(AA09(I).GE.0.) THEN
            F=AA1(I) - AAO(I)
            F9= (AA19(I)-AA09(I))/DTIME
            XK=XAAO(I)-AAO(I)
            XK9=(XAA09(I)-AA09(I))/MBDT
            IF (F9.LE.0..OR.XK9.GE.0.) then
               XMB(I)=0.
               XMB9(I)=0.
            ELSE
               XMB(I)=CMBDT*(-F/XK9+(F9*XK)/(XK9*XK9))
               XMB9(I)=-F9/XK9
            ENDIF
```

```
        ENDIF
  175   CONTINUE
```

Modified adjoint code:

```
        DO 175 I=ISTART,IEND
           IF (AA09(I).GE.0.) THEN
              F9= (AA19(I)-AA09(I))/DTIME
              XK9=(XAA09(I)-AA09(I))/MBDT
              IF (F9.LE.0..OR.XK9.GE.0.) then
                 XMB(I)=0.
                 F=0.
                 XK=0.
              ELSE
                 F=CMBDT*(-XMB(I)/XK9)
                 XK=CMBDT*F9*XMB(I)/(XK9*XK9)
              ENDIF
              XAA0(I)=XK
              AA0(I)=-XK
              AA1(I)=F
              AA0(I)=-F+AA0(I)
           ENDIF
  175   CONTINUE
```

A 13 digit accuracy is obtained.

4.2.9 Implicit redundant calculation

In some physics routines, there are cases containing implicit redundant calculations, which are necessary for the forward model. For example, in the NCEP parameterization scheme which deals with the gravity wave drag, there is a loop which implements the following calculation:

$$DTFAC = min_k\{DTFAC, abs(TMP(k)), k = 1, ksmm1\}$$

where $TMP(k)$ is a function of variables V and T.

The linear code and the adjoint code is shown below:

Example 14:

```
      DO 1610 I=1,IDIMT
         DTFAC (I)=0.
         DTFAC9(I)=1.
1610  CONTINUE
      DO 1620 K=1,KSMM1
      DO 1620 I=1,IDIMT
         IF (T9(I,K).NE.0.) THEN
            TMP=(T9(I,K)*V(I,K)-V9(I,K)*T(I,K))/(2.*DELTIM*RCS*T9(I,K)*T9(I,K))
            TMP9=V9(I,K)/(2.*DELTIM*RCS*T9(I,K))
            IF (TMP9.LT.0) THEN
               TMP =-TMP
               TMP9=-TMP9
            ENDIF
            IF (DTFAC9(I).GT.TMP9) THEN
               DTFAC (I)=TMP
               DTFAC9(I)=TMP9
            ENDIF
         ENDIF
1620  CONTINUE
```

Adjoint code:

```
      DO 1610 I=1,IDIMT
         DTFAC9(I)=1.
```

```
1610  CONTINUE
      DO 16202 K=KSMM1,1,-1
      DO 16202 I=1,IDIMT
         IF (T9(I,K).NE.0.) THEN
            TMP9=V9(I,K)/(2.*DELTIM*RCS*T9(I,K))
            IF (TMP9.LT.0) THEN
               TMP9=-TMP9
            ENDIF
            IF (DTFAC9(I).GT.TMP9) THEN
               TMP=DTFAC (I)
            ELSE
               TMP=0.
            ENDIF
            TMP9=V9(I,K)/(2.*DELTIM*RCS*T9(I,K))
            IF (TMP9.LT.0) THEN
               TMP =-TMP
            ENDIF
            V(I,K)=T9(I,K)*TMP/(2.*DELTIM*RCS*T9(I,K)*T9(I,K))
            T(I,K)=-V9(I,K)*TMP/(2.*DELTIM*RCS*T9(I,K)*T9(I,K))+T(I,K)
         ENDIF
1620  CONTINUE
```

On the surface, we can find nothing wrong with the adjoint code. However, in the case when DTFAC9(I) > TMP9 occurs more than once, the adjoint code will never provide a correct answer. The reason is that in the original code, only the value of TMP at one level on which TMP reaches its maximum is actually used for the later calculation. The TMP calculation on other levels is needed only for finding such a level. Therefore, the correct adjoint code of example 14 is given as below:

Modified adjoint code:

```
      DO 16102 I=1,IDIMT
```

```
         DTFAC9(I)=1.
         KDTFAC9(I)=0
16102 CONTINUE
      DO 16202 K=1,KSMM1
      DO 16202 I=1,IDIMT
         IF (T9(I,K).NE.0.) THEN
            TMP9=V9(I,K)/(2.*DELTIM*RCS*T9(I,K))
            IF (TMP9.LT.0) THEN
               TMP9=-TMP9
            ENDIF
            IF (DTFAC9(I).GT.TMP9) THEN
               DTFAC9(I)=TMP9
               KDTFAC9(I)=K
c              (Save the level on which adjoint calculation is needed)
            ENDIF
         ENDIF
16202 CONTINUE
      DO 16104 I=1,IDIMT
         DTFAC9(I)=1.
16104 CONTINUE
      DO 1620 I=1,IDIMT
         IF (KDTFAC9(I).GT.0) THEN
            K=KDTFAC9(I)
            TMP9=V9(I,K)/(2.*DELTIM*RCS*T9(I,K))
            IF (TMP9.LT.0) THEN
               TMP9=-TMP9
            ENDIF
            IF (DTFAC9(I).GT.TMP9) THEN
               TMP=DTFAC(I)
            ELSE
               TMP=0.
            ENDIF
            TMP9=V9(I,K)/(2.*DELTIM*RCS*T9(I,K))
```

```
          IF (TMP9.LT.0.) THEN
             TMP=-TMP
          ENDIF
          V(I,K)=T9(I,K)*TMP/(2.*DELTIM*RCS*T9(I,K)*T9(I,K))
          T(I,K)=-V9(I,K)*TMP/(2.*DELTIM*RCS*T9(I,K)*T9(I,K))+T(I,K)
       ENDIF
 1620  CONTINUE
```

4.2.10 Adjoint of FFT

For a spectral model, adjoints of both the forward and backward fast Fourier transform (FFT) are needed in the adjoint model. The forward and backward FFT is almost self-adjoint, differing only by a constant factor.

Example 15:

The forward and backward FFTs in the NCEP model:

```
      CALL FFSNFS( BF,WRKFFT(1,1), NFS,WRKFFT(1, NFS+1))
      CALL FFANFA(B(1,1),UQF(1,1),WRKFFT, NFA2)
```

The adjoint of the forward FFT (subroutine FFSNFS) is

```
      CALL FFANFA(BF(1,1),BF(1,1),WRKFFT,NFS)
      DO 2040 K=1,LEVS
      DO 2040 I=1,LONF2
         BF(I,K)=BF(I,K)/RECIP(I)
 2040 CONTINUE
```

and the adjoint of the backward FFT (subroutine FFANFA) is

```
      DO 1040 K=1,LEVS
      DO 1040 I=1,LONF2
         B  (I,K)=BF(I,K)*RECIP(I)
1040  CONTINUE
      CALL FFSNFS(B,WRKFFA(1,1),NFA,WRKFFA(1,NFA+1))
```

where

```
      NFS=LEVS    ! LEVS is the total vertical levels
      NFA=LEVS
      RECIP(1)=1. E0/LONF.  (LONF is the total grid points in longitude)
      RECIP(2)=RECIP(1)
      DO 1009 I=3,LONF
         RECIP(I)=1. E0/LONF2.
1009  CONTINUE
      DO 1010 I=1,LONF
         RECIP(I+LONF)=RECIP(I)
1010  CONTINUE
```

The double-sized array of RECIP is needed due to the fact that a complex number is placed in a real array with the first component representing its real part and the second its imaginary part of the zonal flow, the third and the fourth components for the wave number 1, and so on.

Such adjoint coding construction of FFT works for any type of FFT except that the value of RECIP may vary by a factor of 2.

Finally, we shall emphasize that the adjoint coding by hand is always needed even if one wishes to use an automatic adjoint generator for the adjoint model development. It's true that given a *clean* code, an automatic adjoint generator can automatically produce an adjoint code. However, the code of a complex numerical weather prediction, particularly that of physical parameterization schemes, is never clean. Therefore, even if one specifies the *independent variables* (defined as program input variables with respect to which derivatives are desired), *dependent variables* (output variables whose derivatives are desired), and *active variables* (program variables with which a derivative objective is associated) correctly, which is required for use of an automatic adjoint generator, an automatic adjoint generator does not necessarily produce a correct adjoint code. If this is the case (as often happens), how shall the adjoint model developer be able to find out where the error exists? Extensive adjoint coding experiences are still needed to gain knowledge in making the direct code clean and in effectively debugging and correcting errors in the automatic-generated adjoint code.

Although the automatic adjoint generator is still in its developing stage, it holds great promise for the future of adjoint model development.

Appendix A

Let's show the equivalence between the eqs. (2.49) and (2.43). The variational solution is given by (2.49). With some algebraic manipulations, we can also write:

$$\begin{aligned} x' &= x_b + [\, B^{-1} + H^T O^{-1} H \,]^{-1}\, H^T O^{-1} (\, y - Hx_b \,) \\ &= x_b + [\, B^{-1}(I + BH^T O^{-1} H) \,]^{-1}\, H^T O^{-1} (\, y - Hx_b \,) \\ &= x_b + [\, I + BH^T O^{-1} H \,]^{-1} B\, H^T O^{-1} (\, y - Hx_b \,) \end{aligned} \tag{A1}$$

Let note $Z = BH^T O^{-1}$, then (A1) becomes:

$$\begin{aligned} x' &= x_b + [\, I + ZH \,]^{-1} Z [\, y - Hx_b \,] \\ &= x_b + \left[\, Z^{-1}[\, I + ZH \,] \right]^{-1} [\, y - Hx_b \,] \\ &= x_b + [\, Z^{-1} + H \,]^{-1} [\, y - Hx_b \,] \\ &= x_b + \left[\, [\, I + HZ \,] Z^{-1} \right]^{-1} [\, y - Hx_b \,] \\ &= x_b + Z[\, I + HZ \,]^{-1} [\, y - Hx_b \,] \end{aligned} \tag{A5}$$

And thus:

$$\begin{aligned} x' &= x_b + BH^T O^{-1} [\, I + HBH^T O^{-1} \,]^{-1} [\, y - Hx_b \,] \\ &= x_b + BH^T \left[\, [\, I + HBH^T O^{-1} \,] O \right]^{-1} [\, y - Hx_b \,] \end{aligned} \tag{A6}$$

And finally:

$$x' = x_b + BH^T [\, O + HBH^T \,]^{-1} [\, y - Hx_b \,] \tag{A7}$$

Which is the minimal variance estimate of (2.43).

Preceding page blank

REFERENCES

Anderson, L. A., 1991: The robustness of barotropic unstable modes in a zonally varying atmosphere. *J. Atmos. Sci.* , **48**, 2393-2410.

Bannon, P. R., 1995: Hydrostatic adjustment: Lambs' problem. *J. Atmos. Sci.*, **52**, 1743-1752.

Bennett, A. F., 1992: *Inverse methods in physical oceanography.* Cambridge University, Press, Cambridge.

Bloom, S. C., and S. Shubert, 1990: The influence of Monte-Carlo estimates of model error growth on the GLA OI assimilation system. *Int Symp. of Observations in Meteorology and Oceanography*, Geneva Switzerland, WMO, 467-470.

Boer G., J., 1984: A spectral analysis of predictability and error in an operational forecast system. *Mon. Wea. Rev.*, **112**, 1183-1197.

Buckley, A. G. and A. Lenir, 1983: QN-like variable storage conjugate gradients. *Math. Programming*, **27**, 155-175.

Buizza, R., J. Tribbia,, F. Molteni,, and T. N. Palmer, 1993: Computation of optimal unstable structures for a numerical weather prediction model. *Tellus*, **45A** , 388-407.

Courant, R. and D. Hilbert, 1989: *Methods for Mathematical Physics, Volume I.* Wiley-Interscience, New-York.

Courtier, Ph., J.-N. Thépaut and A. Hollingworth, 1994: A strategy for operational implementation of 4D-VAR using an incremental approach. *Quart. J. R. Met. Soc.*, **120**, pp. 1367-1387.

Courtier, P., 1997: Dual formulation of four dimensional variational assimilation. *Quart. J. R. Met. Soc.*, **123**, 2444-2461.

Dalcher A. and E. Kalnay, 1987: Error growth and predictability in operational ECMWF forecasts. *Tellus* **39A**, 474-491.

Daley, R., 1992, The effect of serially correlated observations and model error on atmospheric data assimilation. *Mon. Wea Rev.*, **120**, 164-117.

Preceding page blank

Dee, D., 1995: On line estimation of error covariance parameters for atmospheric data assimilation. *Mon. Wea Rev.*, **123**, 1128-1145.

Derber, P., 1989: A variational continuous assimilation Technique. *Mon. Wea Rev.*, 117, 2437-2446.

Dee, D., 1995: testing the perfect-model assumption in variational data assimilation. Proceedings of the *Second International WMO Symposium on Assimilation of Ob servations in Meteorology and Oceanography*, Tokyo 13-17 March, 1995. WMO/TD., 49-54.

De Pondeca, M. S. F. V., A. Barcilon and X. Zou, 1998: An adjoint sensitivity study of the efficacy of modal and non-modal perturbations in causing model block onset. *J. Atm. Sci.*, **55**, 2095-2118.

Errico, R., T. Vukicevic and K. Raeder, 1993: Examination of the accuracy of a tangent linear model. *Tellus*, **45A**, pp. 462-477.

Farrell, B., 1982: The initial growth of disturbances in a baroclinic flow. *J. Atmos. Sci.*, **39**, 1663-1686.

Farrell, B., 1989: Optimal excitation of baroclinic waves. *J. Atmos. Sci.*, **46**, 1193-1206.

Frederiksen, J.S. 1997: Adjoint sensitivity and finite-time normal mode disturbances during blocking. *J. Atmos. Sci.*, **54**, 1144-1165.

Gelb, A., 1980: *Applied Optimal Estimation.* The M.I.T press, Cambridge, MA.

Giering, R. and T. Kaminski, 1996: Recipes for adjoint code construction. Internal Report 212 from Max-Plank Institute fur Meteorologie, Hamburg, Germany. 35pp.

Gill, P. E. and W. Murray, 1979: Conjugate-gradient methods for large-scale nonlinear optimization. *Tech. Report SOL 79-15*, Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, CA.

Gill, P. E., W. Murray and M. H. Wright, 1981: *Practical Optimization.* Academic Press, 401pp.

Goldhirsch, I., Orszag, S.A. and Maulik, B.K. 1987: An efficient method for computing leading eigenvalues and eigenvectors of large asymmetric matrices. *J. Sci. Comput.* , **2**, 33-58.

Golub, G. H. and Van Loan, C.F. 1989: Matrix Computations, The Johns Hopkins University Press, 642pp.

Grell, G. A., Dudhia, J. and Stauffer D. R. 1994. A description of the fifth-generation Penn State/NCAR mesoscale model (MM5). *NCAR Technical Note* , NCAR/TN-398 + STR, National Center for Atmospheric Research, Boulder, CO, 138 pp.

Hall, M. C. G. and D. G. Cacuci, 1983: Physical interpretation of the adjoint functions for sensitivity analysis of atmospheric models. *J. Atmos. Sci.*, **40**, 2537-2546.

Jazwinsky, A. H., 1970: *Stochastic process and filtering theory.* Mathematic in Science and Engineering Vol 64, Academic Press, New York, 376p.

Kuo, Y.-H., X. Zou, and Y.-R. Guo, 1995: Variational assimilation of precipitable water using a nonhydrostatic mesoscale adjoint model. Part I: Moisture retrieval and sensitivity experiments. *Mon. Wea. Rev.*, **124**, 122-147.

Kuo, Y.-H., X. Zou, and W. Huang , 1996: The Impact of GPS Data on the Prediction of an Extratropical Cyclone: An Observing System Simulation Experiment. *J. Dyn. Atmos. Ocean.*, (in press)

Lacarra J. F. and O. Talagrand, 1988: Short range evolution of small perturbations in a barotropic model. *Tellus*, **40A**, pp. 81-95.

Liu, D. C. and Nocedal, J. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming* **45**, 503-528.

Lorenz, E. N., 1963: Deterministic non-periodic flow. *J. Atmos. Sci.*, **20**, 130-141.

Lorenz, E. N., 1965: A study of the predictability of a 28-variable atmospheric model. *Tellus*, **17**, 321-333.

Molteni, F., Buizza, R., Palmer, T. N. and Petroliagis, T., 1996: The ECMWF ensemble prediction system: Methodology and validation. *Quart. J. Roy. Meteor. Soc.*, **112**, 73-119.

Navon, I. M., Zou, X., Derber, J. and Sela, J. 1992. Variational data assimilation with an adiabatic version of the NMC spectral model. *Mon. Wea. Rev.* **120**, 1433-1446.

Noble, B., and Daniel, J. W. 1977: Applied Linear Algebra, Prentice-Hall Inc., 477pp.

Palmer, T. N., 1995: Predictability of the atmospheres and oceans: from days to decades. In *ECMWF seminar proceedings on Predictability*, Vol I, 83-140. ECMWF, Reading UK.

Palmer, T. N., R. Gelaro, J. Barkmeijer and R. Ruizza, 1998: Singular vectors, metrics and adaptive observations. *J. Atmos. Sci.*, **55**, 633-653.

Pu, Z.-X., E. Kalnay, J. G. Sela, and I. Szunyogh, 1997: Sensitivity of forecast errors to initial conditions with a quasi-inverse linear method. *Mon. Wea. Rev.*, **25**, 2479-2503.

Rabier, F., E. Klinker, P. Courtier, and A. Hollingsworth, 1996: Sensitivity of forecast errors to initial conditions. *Q. J. R. Meteorol. Soc.*, **122**, 121-150.

Shanno, D. F. and K. H. Phua, 1980: Remark on algorithm 500 — a variable method subroutine for unconstrained nonlinear minimization. *ACM Trans. Math. Software*, **6**, 618-622.

Simmons, A. J., 1995: The skill of 500hPa height forecasts. In *ECMWF seminar proceedings on Predictability*, Vol I, 19-68. ECMWF, Reading UK.

Talagrand, O., 1992: Data Assimilation Problems. in Proceedings, NATO Advanced Study Institute, Energy and Water Cycles in the Climate System, (Glüsscksburg Germany, October 1991), E. Raschke and D. Jacob, eds, Forschungszentrum Geesthacht, Geesthacht, Germany, 187-213.

Thépaut, J. N. and Courtier, P. 1991. Four-dimensional variational data assimilation using the adjoint of a multilevel primitive equation model. *Q. J. R. M. S.* **117**, 1225-1254.

Toth, Z, 1991: Estimation of atmospheric predictability by circulation analogs. *Mon. Wea. Rev.*, 119, 65-72.

Toth, Z. and E. Kalnay, 1995: Ensemble forecasting at NCEP. In *ECMWF seminar proceedings on Predictability*, Vol II, 39-60. ECMWF, Reading UK.

Mon. Wea. Rev., (submitted)

Wergen, W. 1992: The effect of model errors in variational data assimilation. Tellus 44A, 297-313.

Zou, X., Navon, I. M., Berger, M., Phua, Paul K. H., Schlick, T. and Le Dimet, F. X. 1993a. Numerical experience with limited-memory quasi-Newton and truncated Newton methods. *SIAM J. on Optimization* **3**, 582-608.

Zou, X., A. Barcilon, I. M. Navon, J. Whitaker, and D. G. Cacuci, 1993b: An adjoint sensitivity study of blocking in a two-layer isentropic model. *Mon. Wea. Rev.*, **121**, 2833-2857.

Zou, X., Kuo, Y.-H. and Guo, Y.-R. 1995. Assimilation of atmospheric radio refractivity using a nonhydrostatic adjoint model. *Mon. Wea. Rev.* **123**, 2229-2249.

Zou, X., 1996: Tangent linear and adjoint of "on-off" processes and their feasibility for use in four-dimensional variational data assimilation. *Tellus*, **49A**, 3-31.

Zou, X. and Kuo, Y.-H., 1996: Rainfall assimilation through an optimal control of initial and boundary conditions in a limited-area mesoscale model. *Mon. Wea. Rev.*, **124**, 2859-2882.

Zou, X and W. Huang, 1997: The MM5 adjoint modeling system: User guide.

Zou, X, F., Y.-H. Kuo, Y.-R. Guo and W. Huang, 1997: Development and application of the MM5 adjoint modeling system. *Submitted to Mon. Wea. Rev.*

Zupanski, D., 1996: A general weak constraint applicable to 4DVAR data assimilation systems. *Submitted to Mon. Wea Rev.*

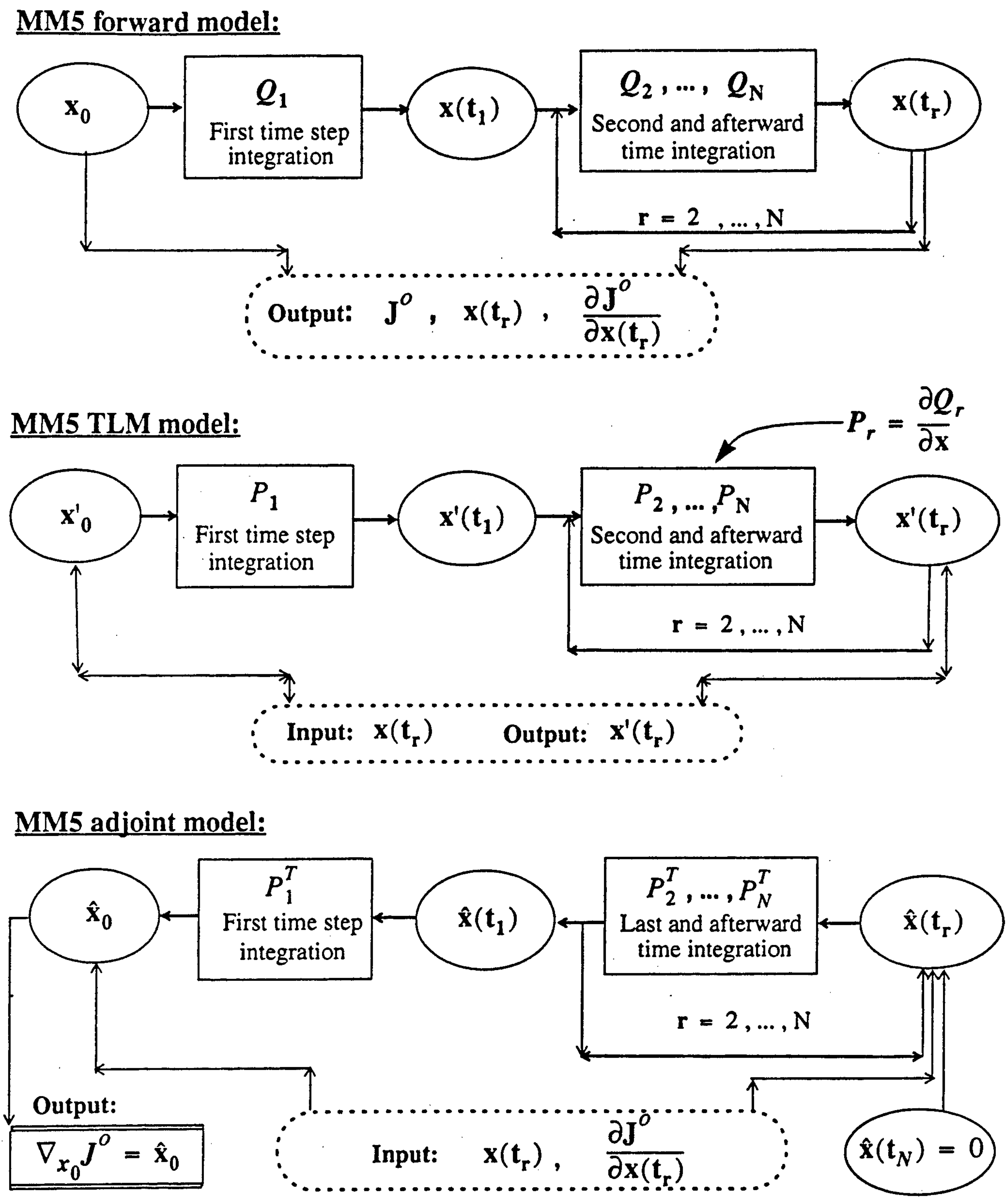
MM5 forward model:
x_0
Q_1 First time step integration
$x(t_1)$
$Q_2, \dots, Q_N$ Second and afterward time integration
$x(t_r)$
$r = 2, \dots, N$
Output: J^o, $x(t_r)$, $\frac{\partial J^o}{\partial x(t_r)}$
MM5 TLM model:
$P_r = \frac{\partial Q_r}{\partial x}$
x'_0
P_1 First time step integration
$x'(t_1)$
$P_2, \dots, P_N$ Second and afterward time integration
$x'(t_r)$
$r = 2, \dots, N$
Input: $x(t_r)$ Output: $x'(t_r)$
MM5 adjoint model:
$\hat{x}_0$
P_1^T First time step integration
$\hat{x}(t_1)$
$P_2^T, \dots, P_N^T$ Last and afterward time integration
$\hat{x}(t_r)$
$r = 2, \dots, N$
Output: $\nabla_{x_0} J^o = \hat{x}_0$
Input: $x(t_r)$, $\frac{\partial J^o}{\partial x(t_r)}$
$\hat{x}(t_N) = 0$

Figure 3.1: Flow chart of the MM5 forward, TLM and adjoint models

Preceding page blank

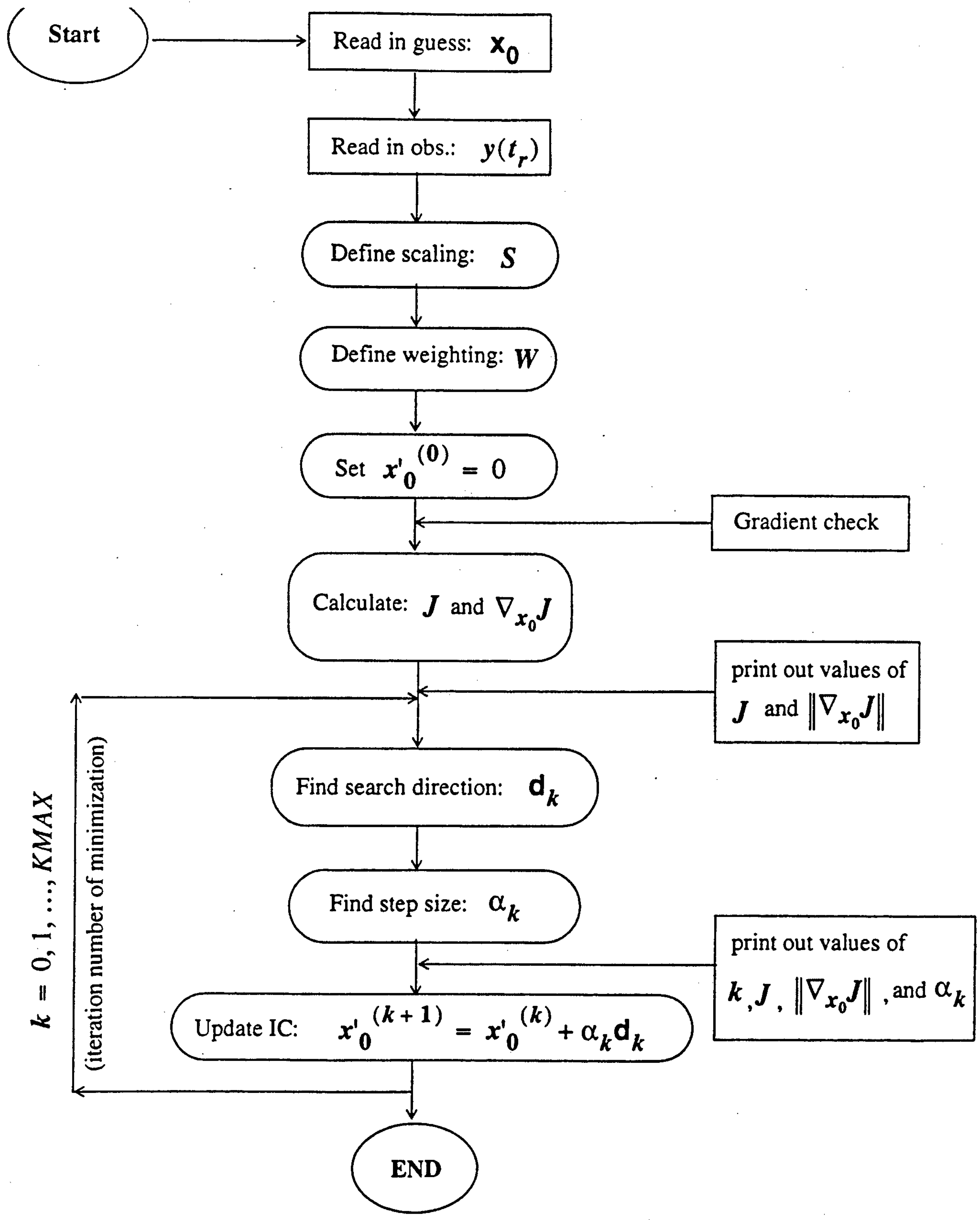
Start
Read in guess: $\mathbf{x}_0$
Read in obs.: $y(t_r)$
Define scaling: S
Define weighting: W
Set $x'^{(0)}_0 = 0$
Gradient check
Calculate: J and $\nabla_{x_0} J$
print out values of J and $\|\nabla_{x_0} J\|$
Find search direction: $\mathbf{d}_k$
Find step size: α_k
print out values of k, J, $\|\nabla_{x_0} J\|$, and α_k
Update IC: $x'^{(k+1)}_0 = x'^{(k)}_0 + \alpha_k \mathbf{d}_k$
$k = 0, 1, \ldots, KMAX$
(iteration number of minimization)
END

Figure 3.2: Flow chart of the main minimization routine.